



Snowflake Fundamentals

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



INTRODUCTIONS

- Name
- Company & Role
- Experience with Snowflake
 - What do you already know?
 - What do you want to learn?
- Tell us something about yourself



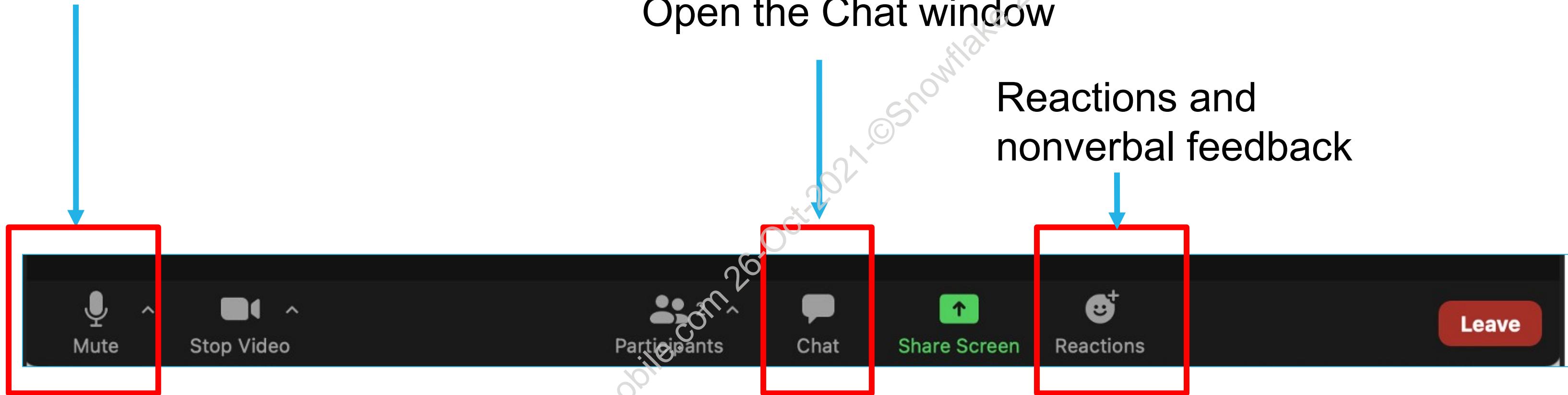
REMOTE ATTENDEE CONTROLS

- Hover over your Zoom interface to display the control panel at the bottom



CONTROL PANEL

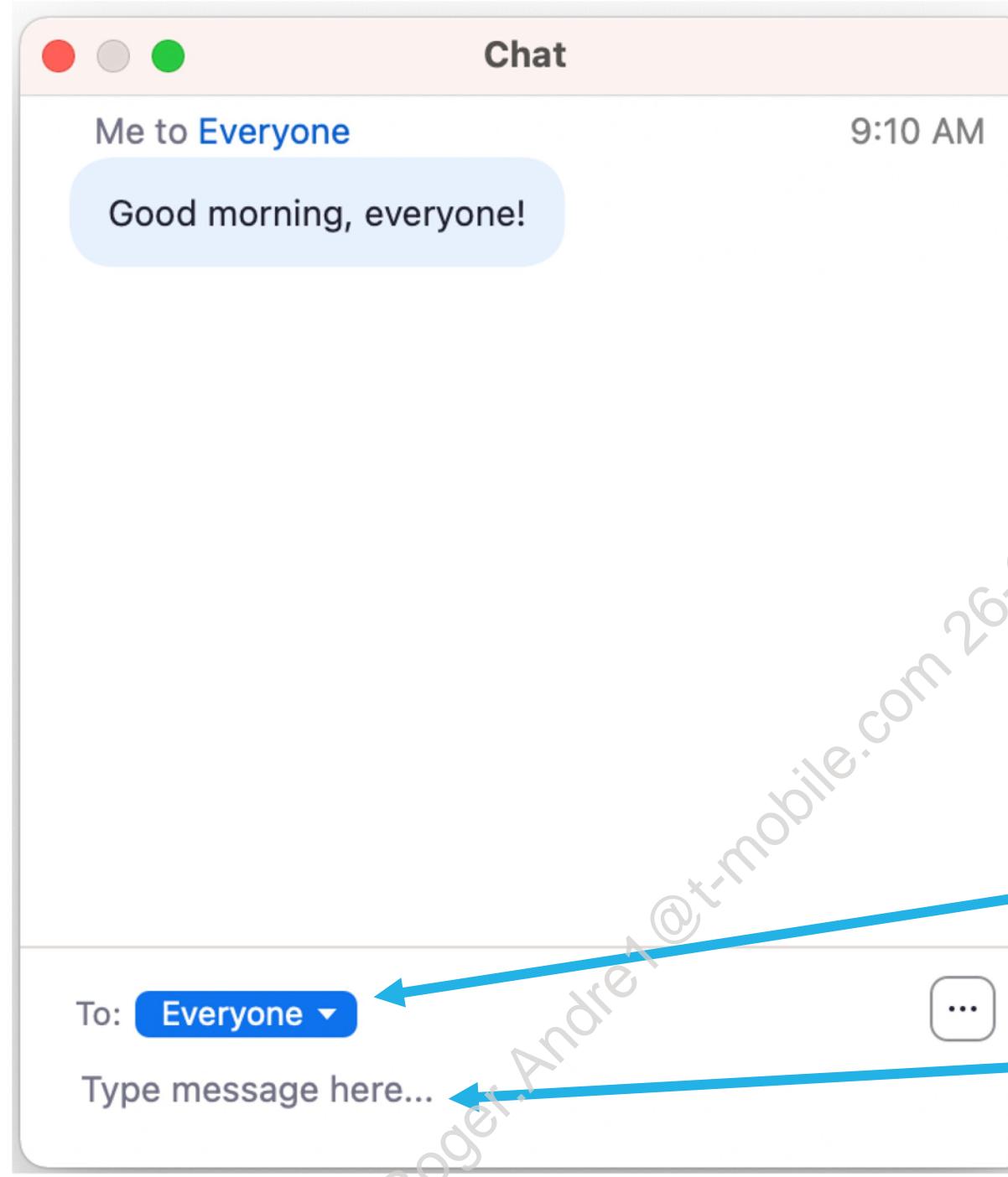
Mute/Unmute



Open the Chat window

Reactions and
nonverbal feedback

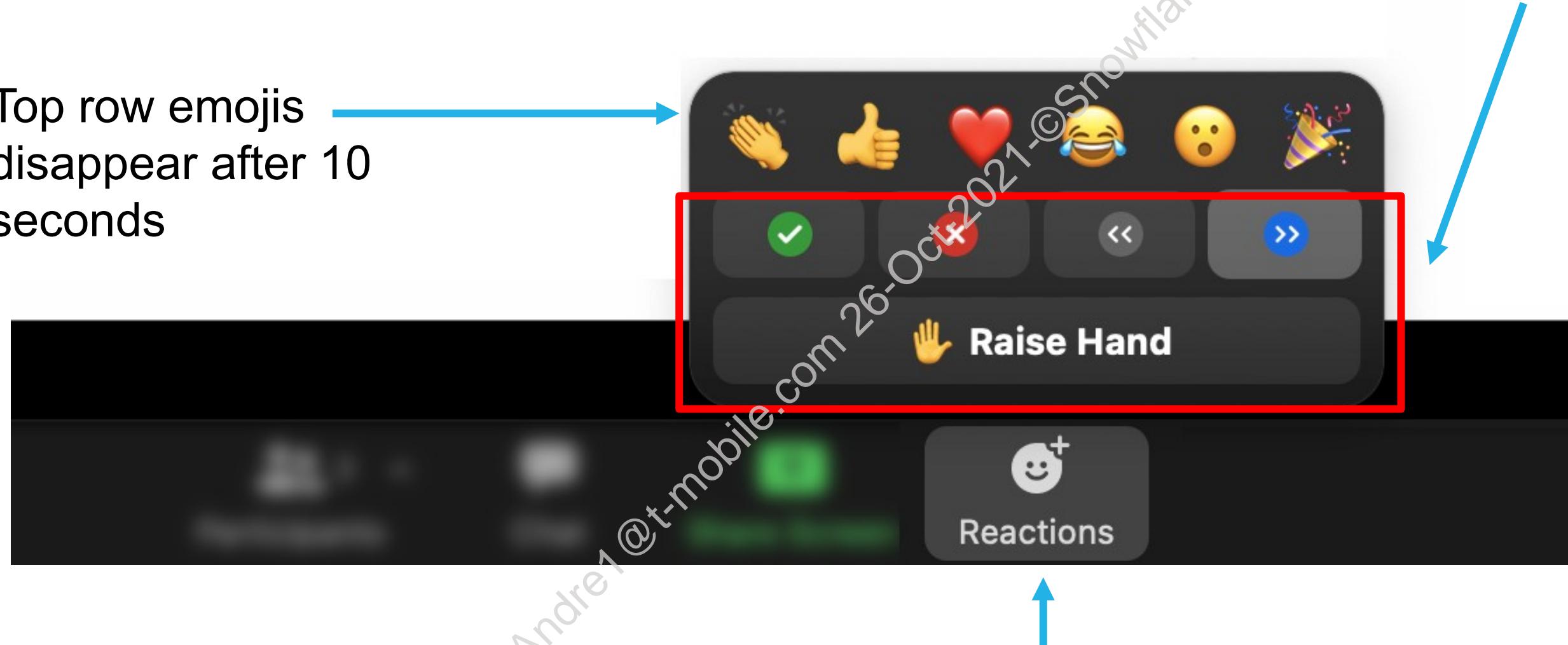
CHAT INTERFACE



1. Use drop-down to choose recipient
2. Click to type your message and press Enter

REACTIONS AND NONVERBAL FEEDBACK

Top row emojis
disappear after 10
seconds



Use bottom two rows to raise your hand,
click Yes/No, etc. (click again to cancel)

COURSE OBJECTIVES

By the end of this course you will be able to set up and administer Snowflake to your users. You will also be able to effectively use snowflake to get business value out of your data. You will be able to:

- Configure and manage account parameters
- Configure users and roles with appropriate privileges
- Load and transform data in Snowflake
- Query data, and size virtual warehouses for different use cases
- Work with semi-structured data in Snowflake
- Understand Snowflake's micro-partitioning and how it contributes to query optimization
- Use Time Travel and failsafe storage to recover data and provide continuous data protection
- Use zero-copy cloning to provide groups with different use cases access to the same data
- Use Data Sharing to send your data in real time to customers, partners or other company accounts



COURSE AGENDA

Overview & Architecture

Clients, Connectors, & Ecosystems

Snowflake Caching

SQL Support in Snowflake

Data Loading and Unloading

Tasks

Snowflake Functions

Managing Security

Access Control and User Management

Semi-Structured Data

Continuous Data Protection

Data Sharing

Performance & Concurrency

Account & Resource Management

Preview Features

Fundamentals Recap



LAB EXERCISE

10 minutes

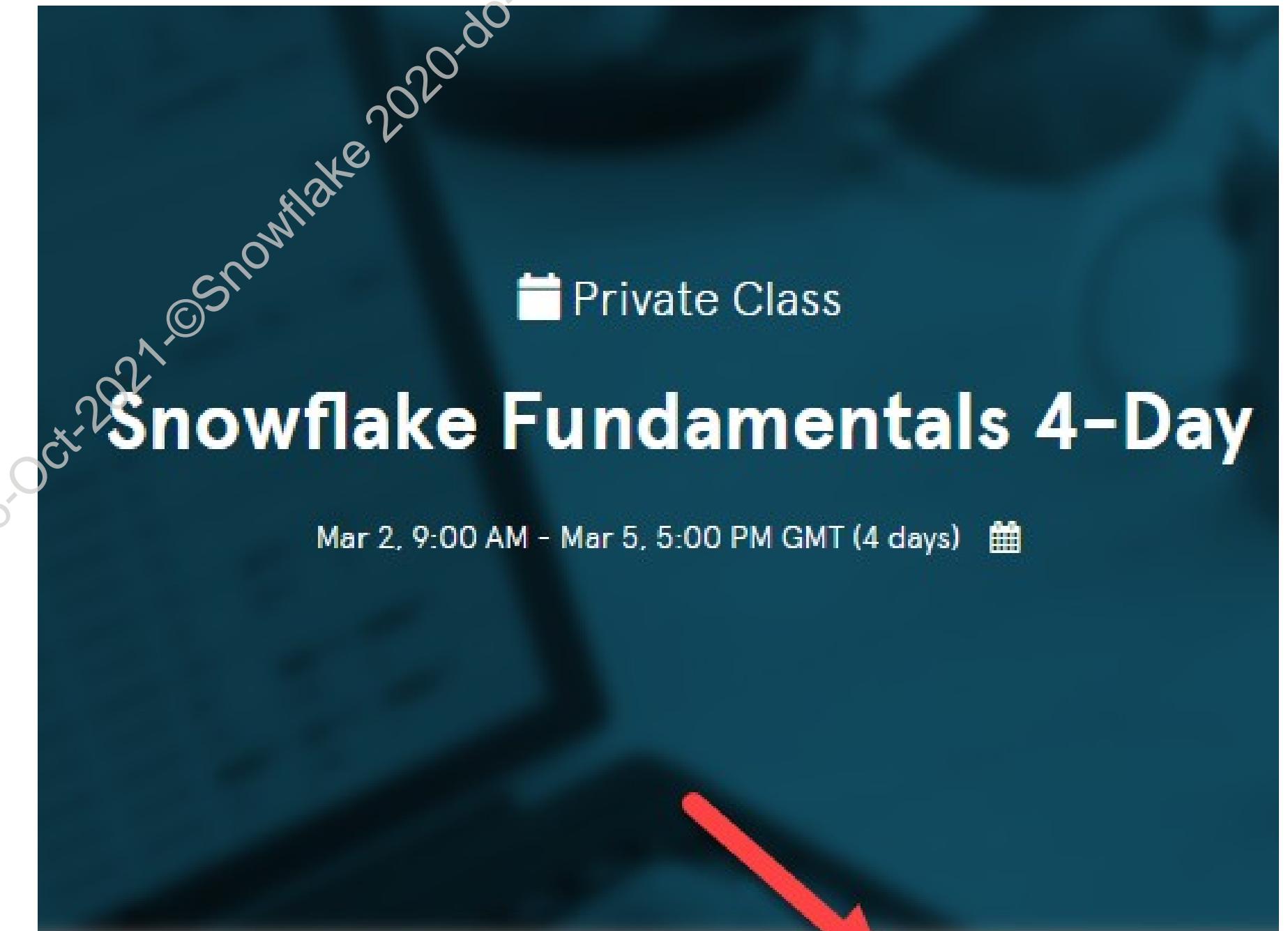
Introduction - Prepare for Class

- Log in to Snowflake Training Portal
 - <https://training.snowflake.com/login.html>
- Instructor will provide you access to a Snowflake account
 - Lab access is available for 30 days from the first day of class.



ACCESSING CLASS FILES

- Login to the class website
 - This is the same website you accessed to get into the remote training.
- Click on the "Resources" link.



ACCESSING CLASS FILES

- This screen shows links to the 3 documents you will need for class.
 - Workbook
 - The lab workbook with the instructions for the class labs.
 - Lab Files
 - A .zip file with an SQL file for each lab. This will need to be opened and copied into a folder.
 - Course Slides
 - A PDF file with the training material for the class.
- Click on each link to open the files.
- Notice the link to the Class Survey.

The screenshot shows a web browser displaying a class details page for a "Snowflake Fundamentals 4-Day" course. The page includes the following elements:

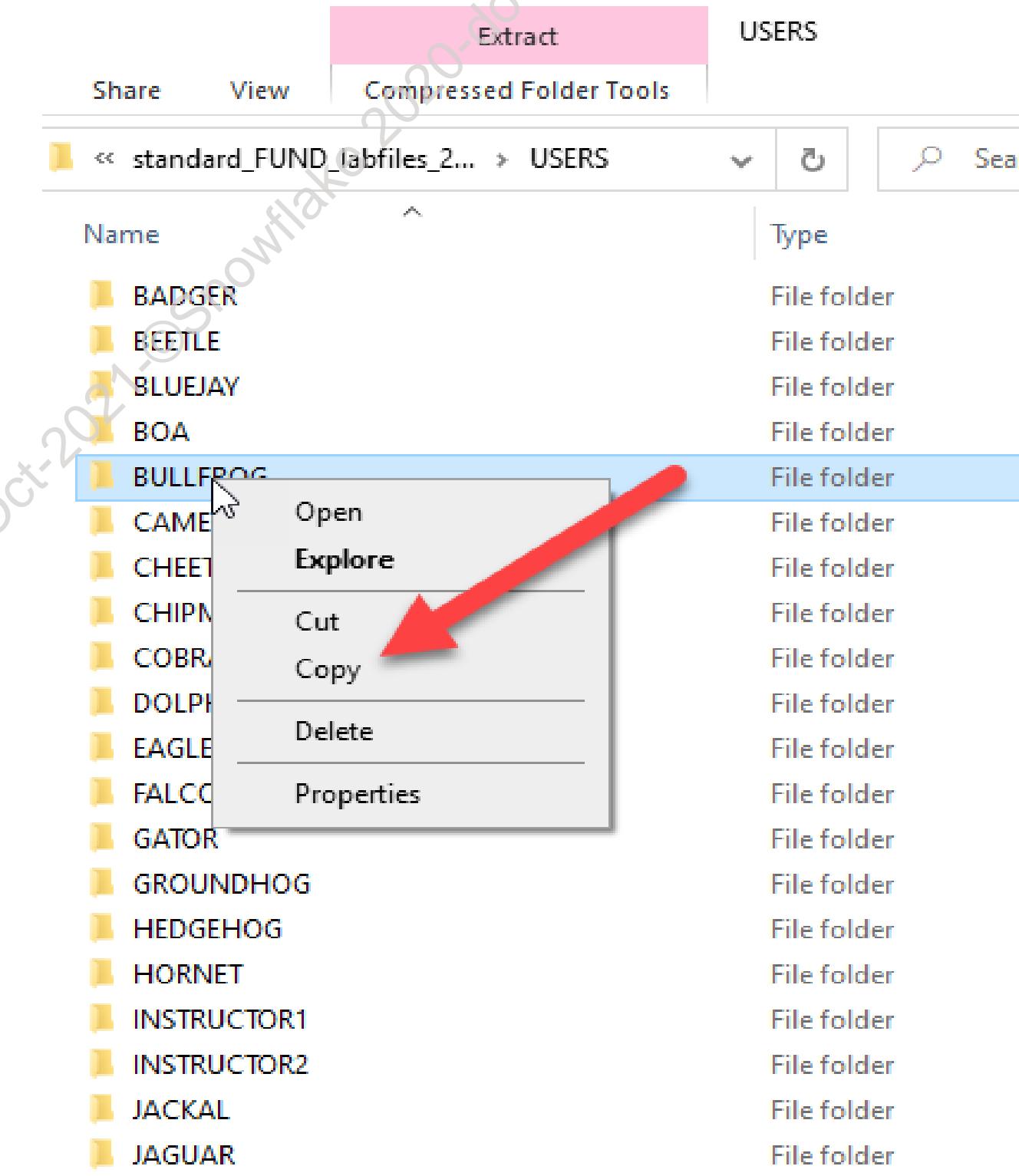
- Header:** ke Dashboard Public Schedule Documentation Support Cart
- Title:** Snowflake Fundamentals 4-Day
- Date:** Mar 2, 9:00 AM - Mar 5, 5:00 PM GMT (4 days)
- Class Type:** Private Class
- Navigation:** Overview (selected), Instructors (1), Resources (4)
- Resources:** Four items listed:
 - Snowflake Post Class Survey
 - Fundamentals 4Day Course LabFiles 2.52MB ZIP
 - Fundamentals 4Day Course WorkBook 1.65MB PDF
 - Fundamentals 4Day Course Slides 13.41MB PDF

Red arrows point from the list on the left to the corresponding resource cards on the right, highlighting the "Fundamentals 4Day Course Slides" and "Fundamentals 4Day Course WorkBook".



ACCESSING CLASS FILES

- The lab files contains a copy of the labs for every student.
- You only need to copy your labs from the file.
- Open the lab file and scroll to
 - /standard_FUND_labfiles###/USERS
- Find the animal's name assigned to you by your instructor.
- Copy just that folder to a folder on your local machine.
- This will be faster than decompressing the whole zip file.
 - For example, if your class name was "bullfrog", just copy the one folder to a local folder on your machine.



OVERVIEW AND ARCHITECTURE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Why Snowflake?
- Snowflake Structure
- Cloud Services Layer
- Data Storage Layer
- Compute Layer

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

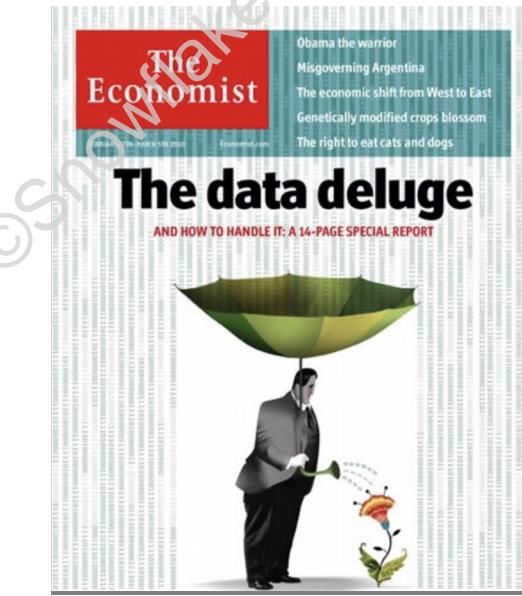


WHY SNOWFLAKE?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

DATA

THE WORLD'S MOST VALUABLE RESOURCE



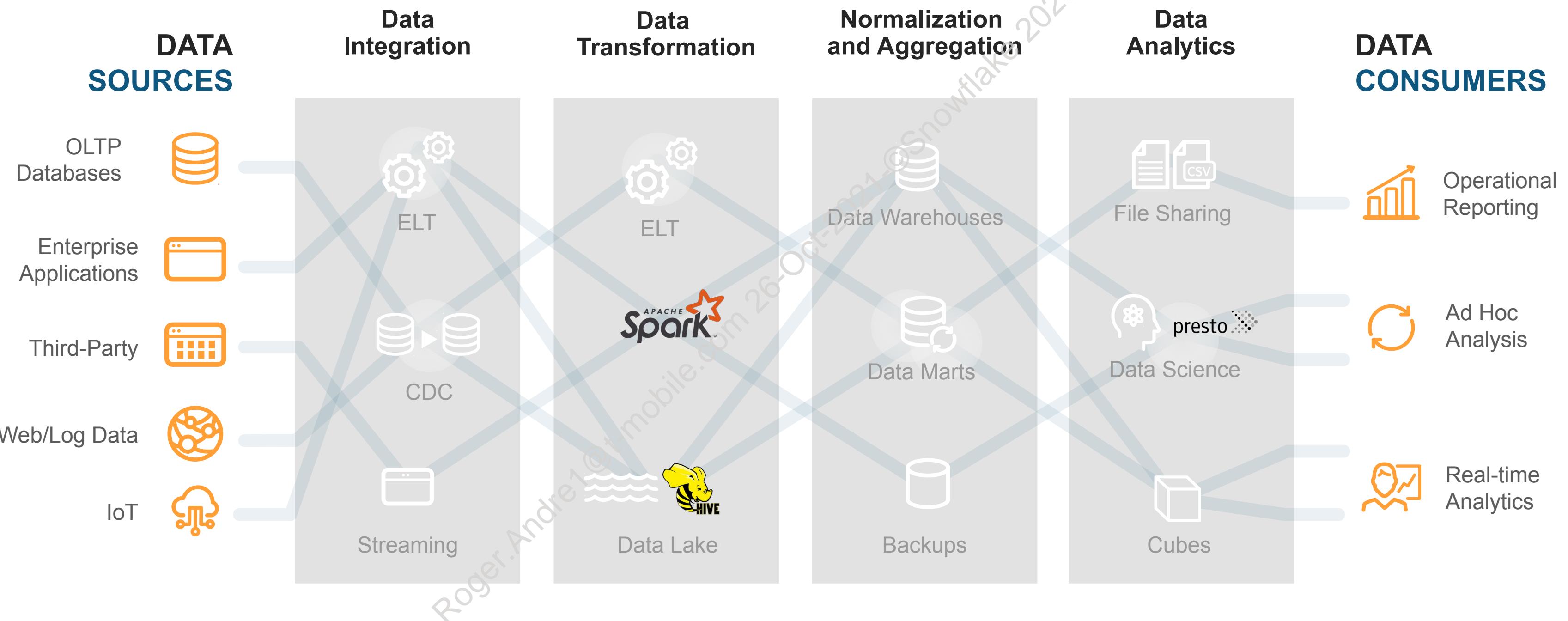
“The rapid rise of gathered/analyzed digital data is often core to the holistic success of the fastest growing and most successful companies of our time around the world.”

– Mary Meeker, Bond Capital

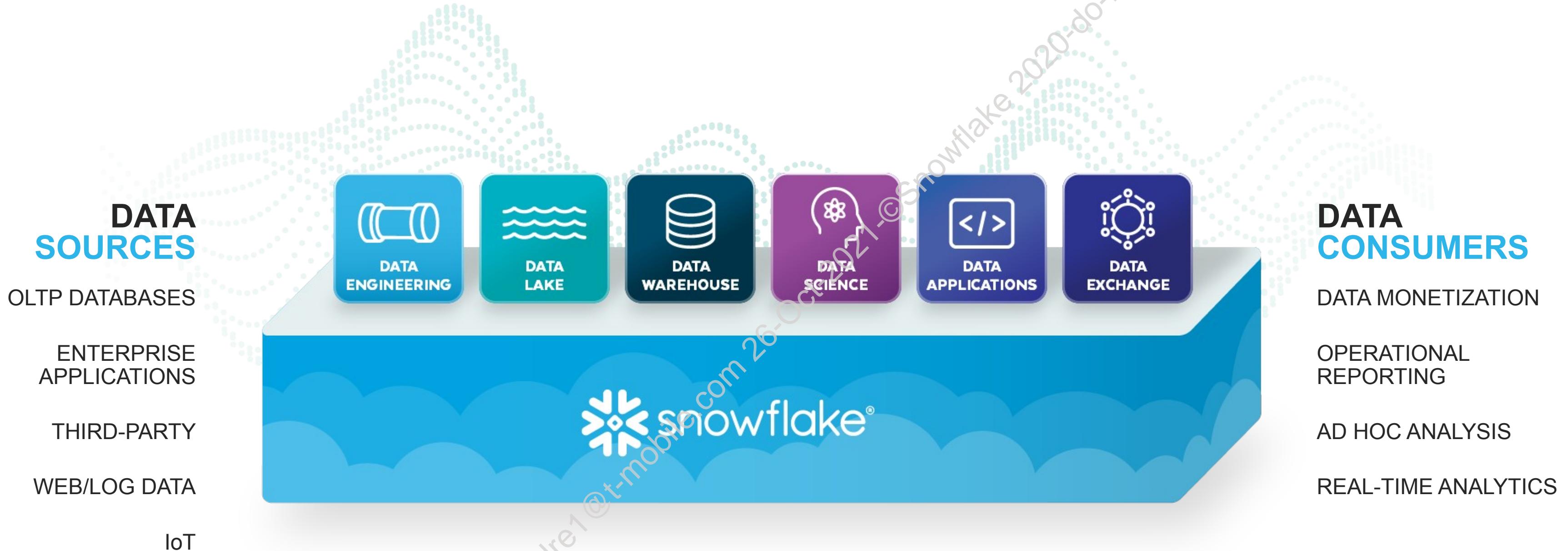


TRADITIONAL DATA ARCHITECTURE

COMPLEX, COSTLY, CONSTRAINED

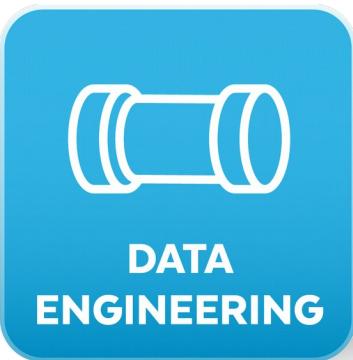


SNOWFLAKE CLOUD DATA PLATFORM

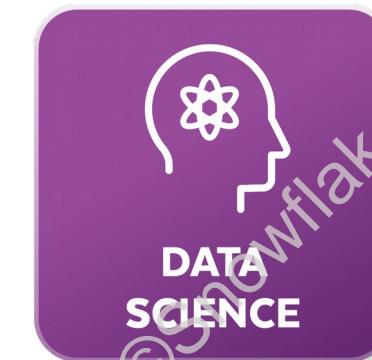


SINGLE GLOBAL PLATFORM

ONE COPY OF DATA, MANY WORKLOADS



Simple, reliable data pipelines
in the language of your choice



Simple data preparation for modeling
with your framework of choice



Access, performance, and
security for your data lake



Build data-intensive applications
without operational burden



Analytics at scale on all your
data with zero administration



Share and collaborate on live data
across your business ecosystem



WHY A CLOUD DATA PLATFORM?

	On Premises EDW	1st Gen Cloud EDW	Data Lake, Hadoop	Cloud Data Platform
All Data	✗	✗	✓	✓
All Users	✗	✗	—	✓
Fast Performance	✓	✓	✗	✓
Easy to use	✓	✓	✗	✓



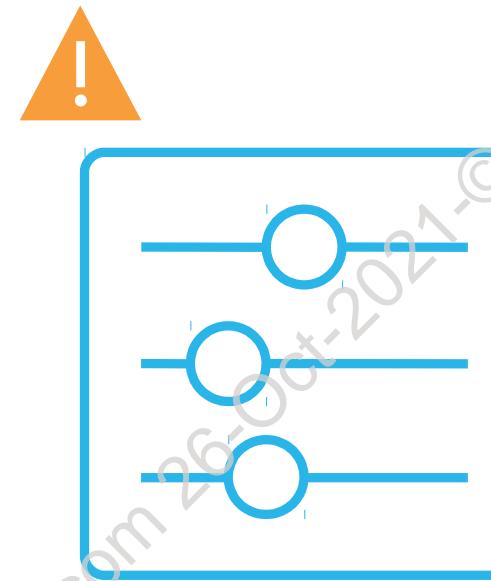
HISTORICAL PERFORMANCE ASSUMPTIONS

Limited, fixed resources



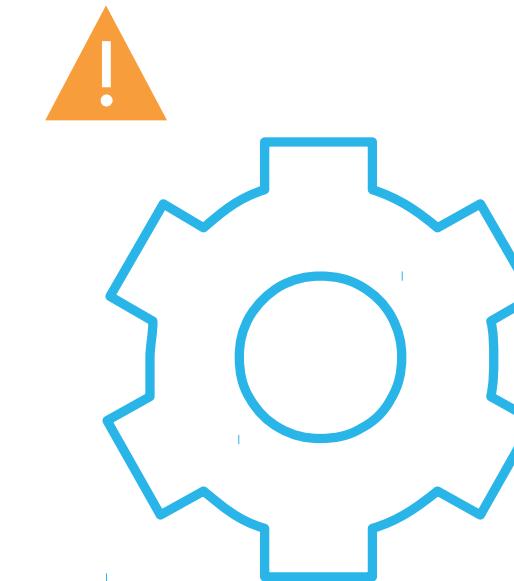
Resources are fixed, so must be sized to the maximum load and protected from overuse.

Tune for performance



Performance issues must be dealt with through tuning, indexing, re-partitioning, and other “knob” turning.

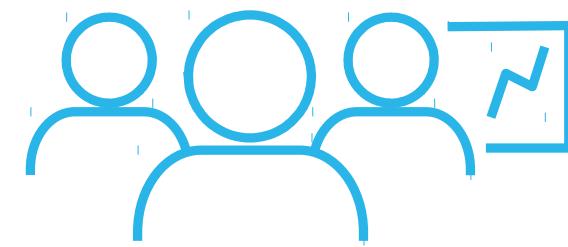
Manual upkeep



Partitioning, indexing, and other performance tuning is manual, requiring ongoing maintenance over time.

SNOWFLAKE APPROACH TO PERFORMANCE

Workload Isolation



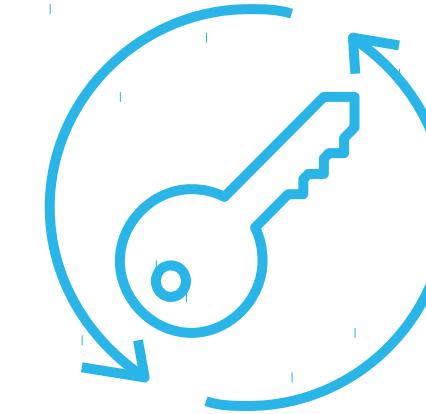
Unlimited compute clusters serve a diverse array of workloads, on-demand.

Simplicity



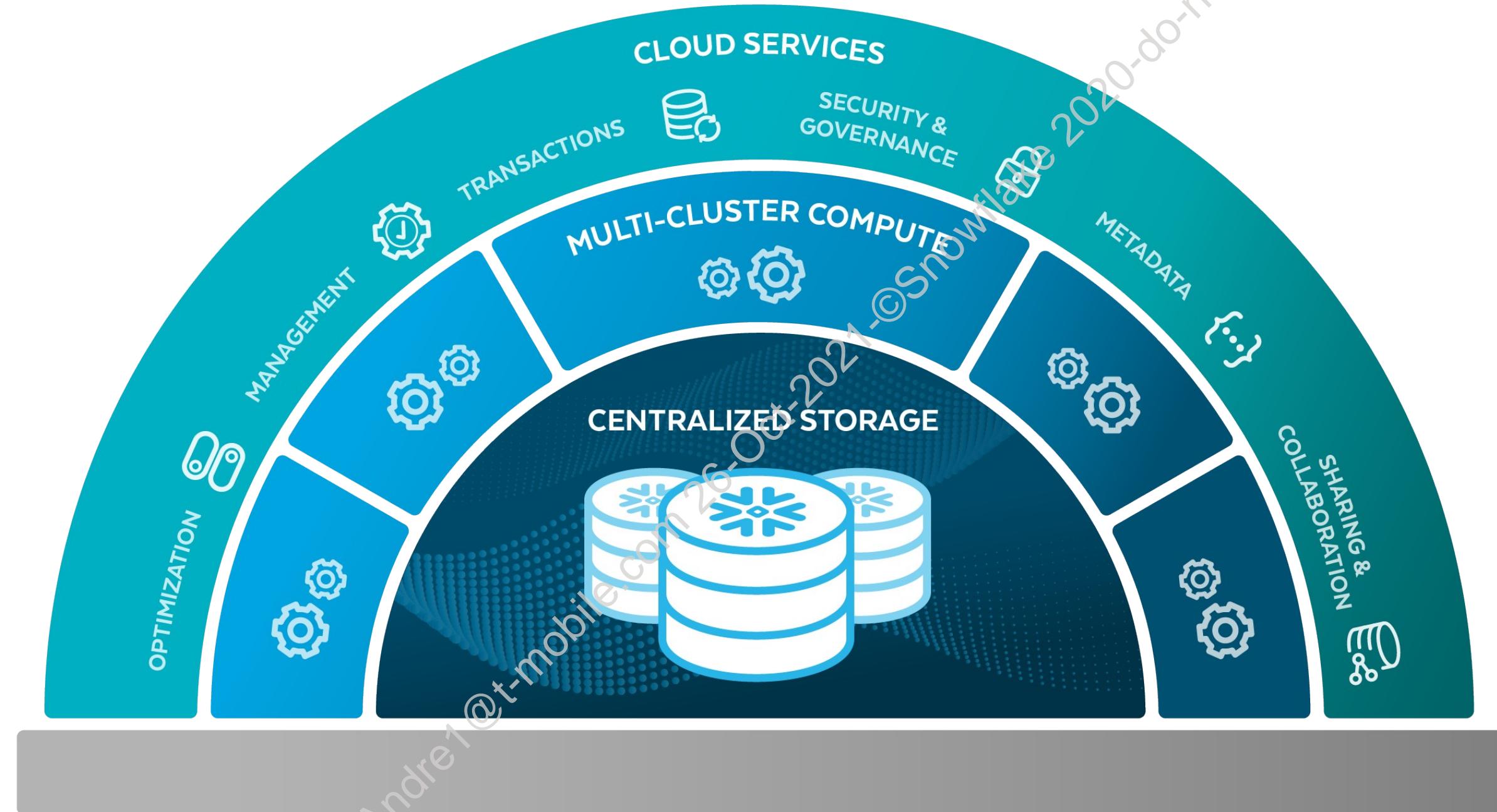
The Snowflake service self-tunes. The user has the option of using materialized views and cluster keys to enhance performance if needed.

Automation

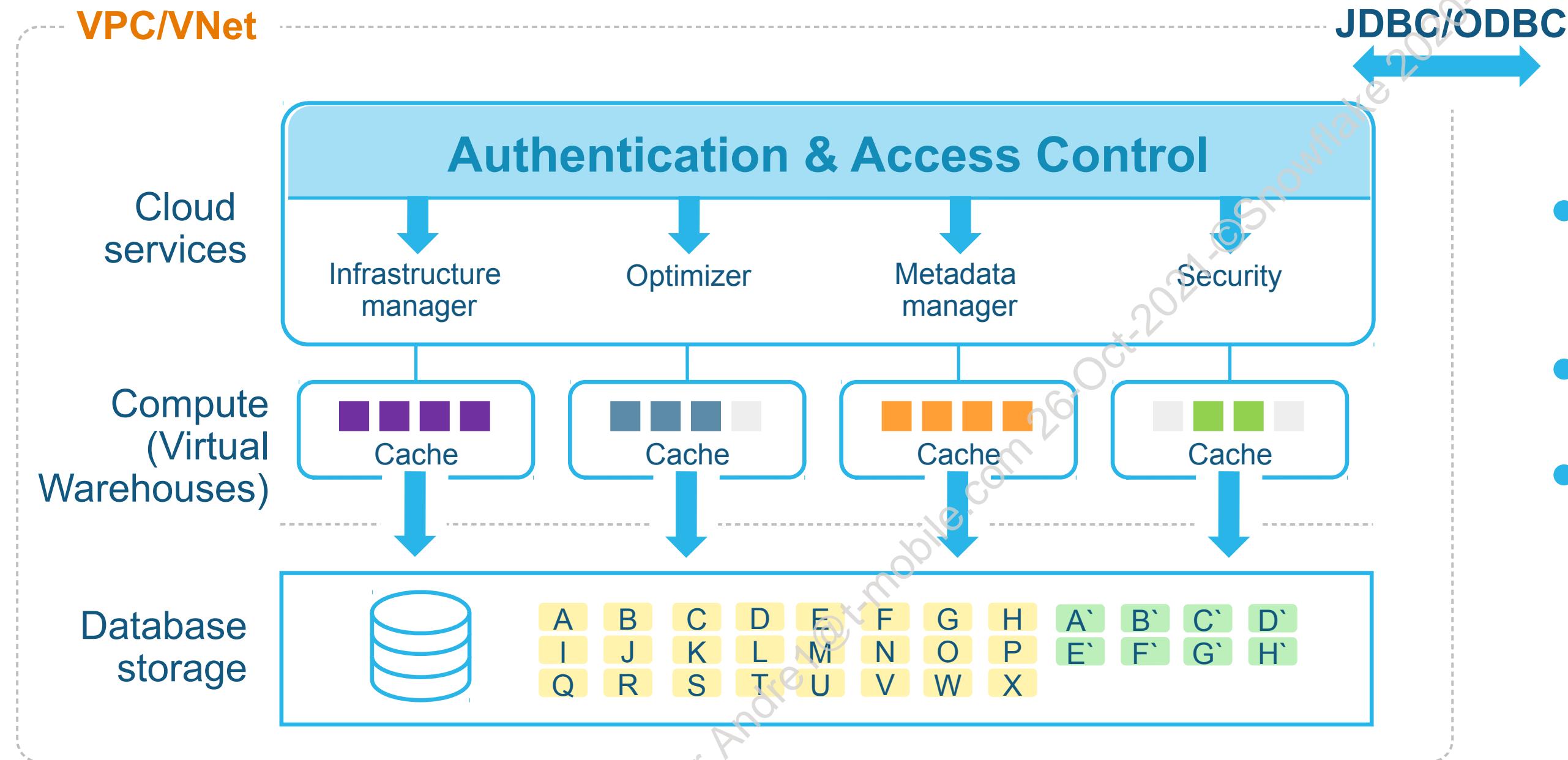


All of the performance processes in Snowflake were designed to run or maintain themselves.

SNOWFLAKE ARCHITECTURE



MULTI-CLUSTER SHARED DATA ARCHITECTURE



- Storage decoupled from compute
- All data in one place
- Dynamically combine storage and compute

FUNCTIONAL ARCHITECTURE



Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



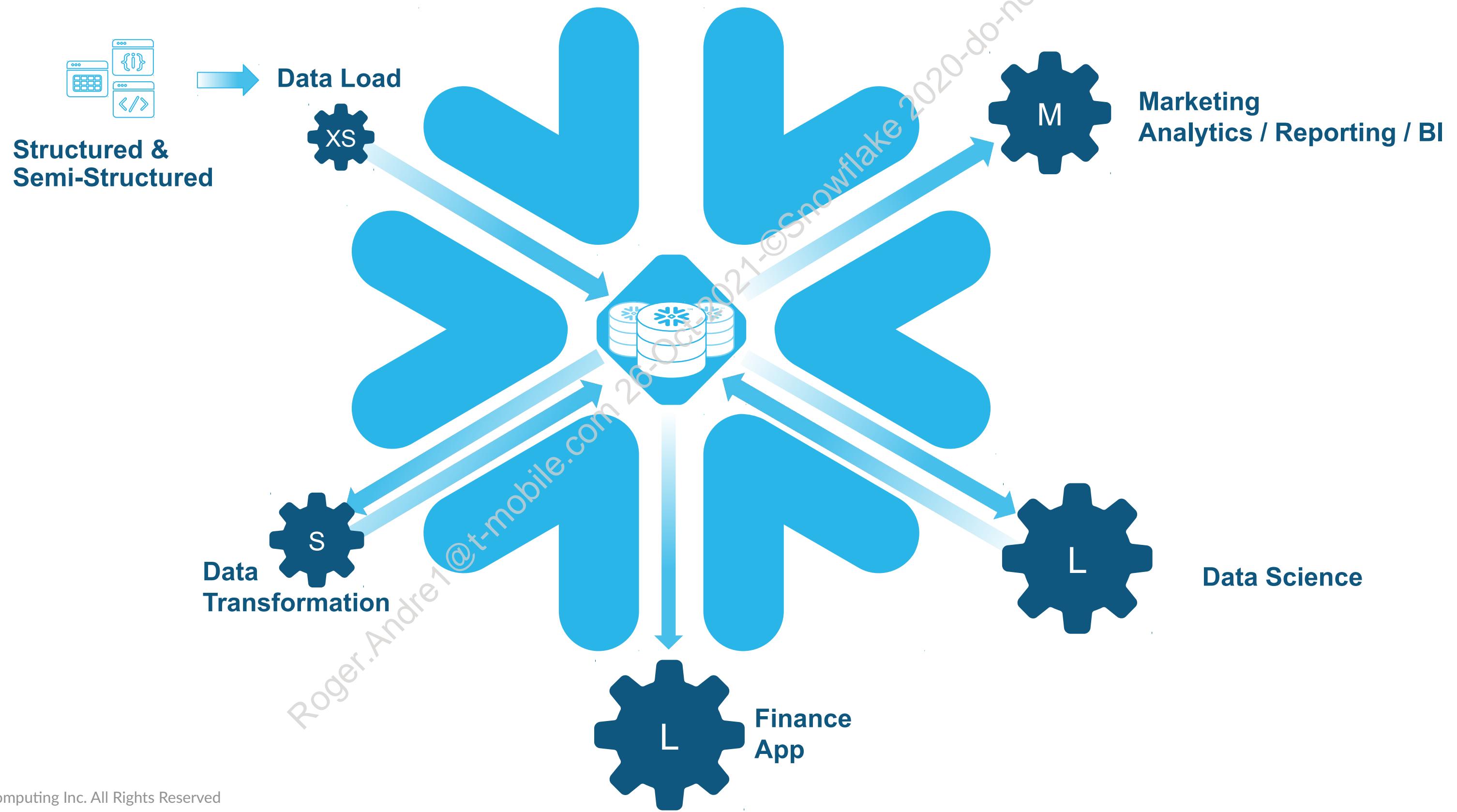
FUNCTIONAL ARCHITECTURE



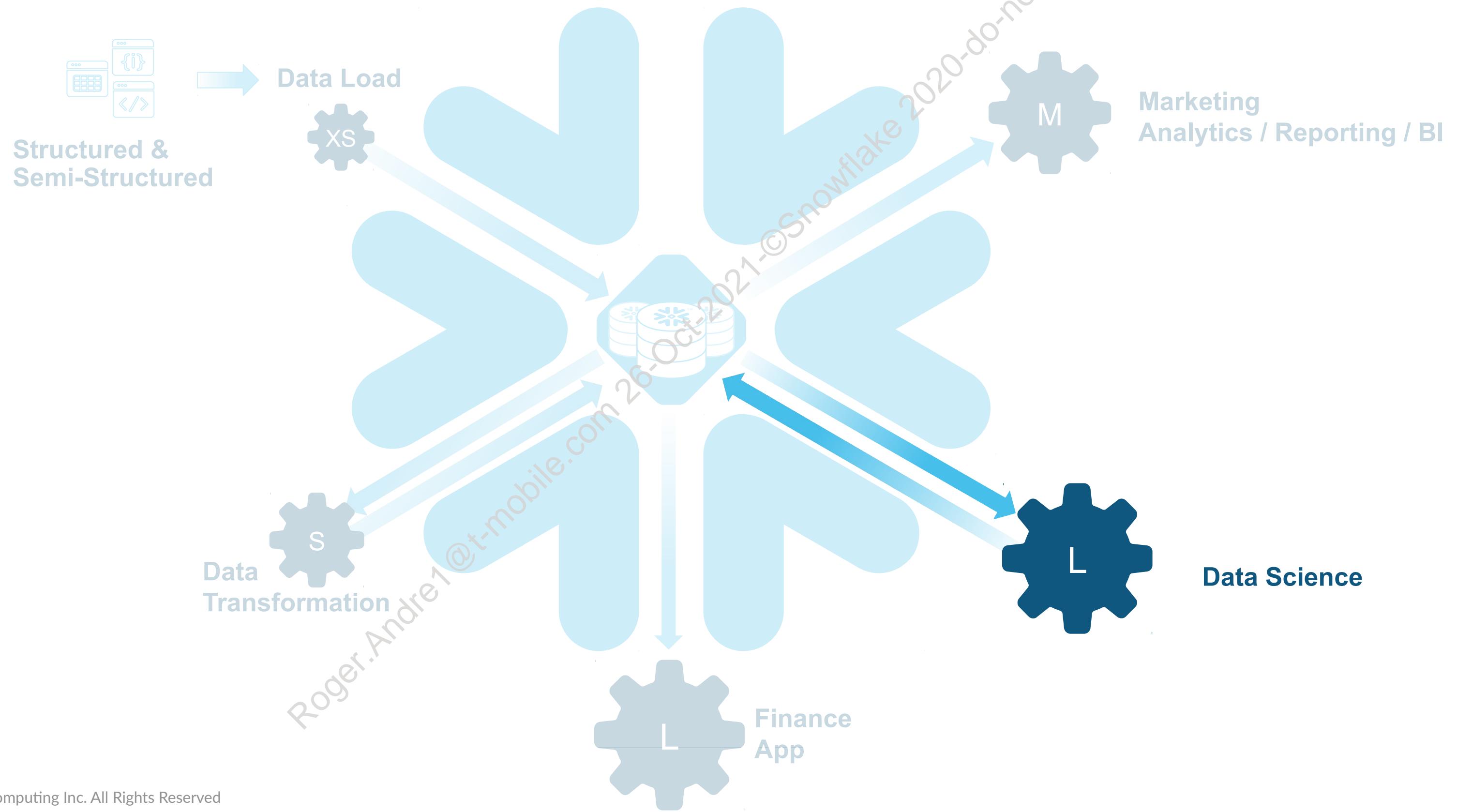
Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



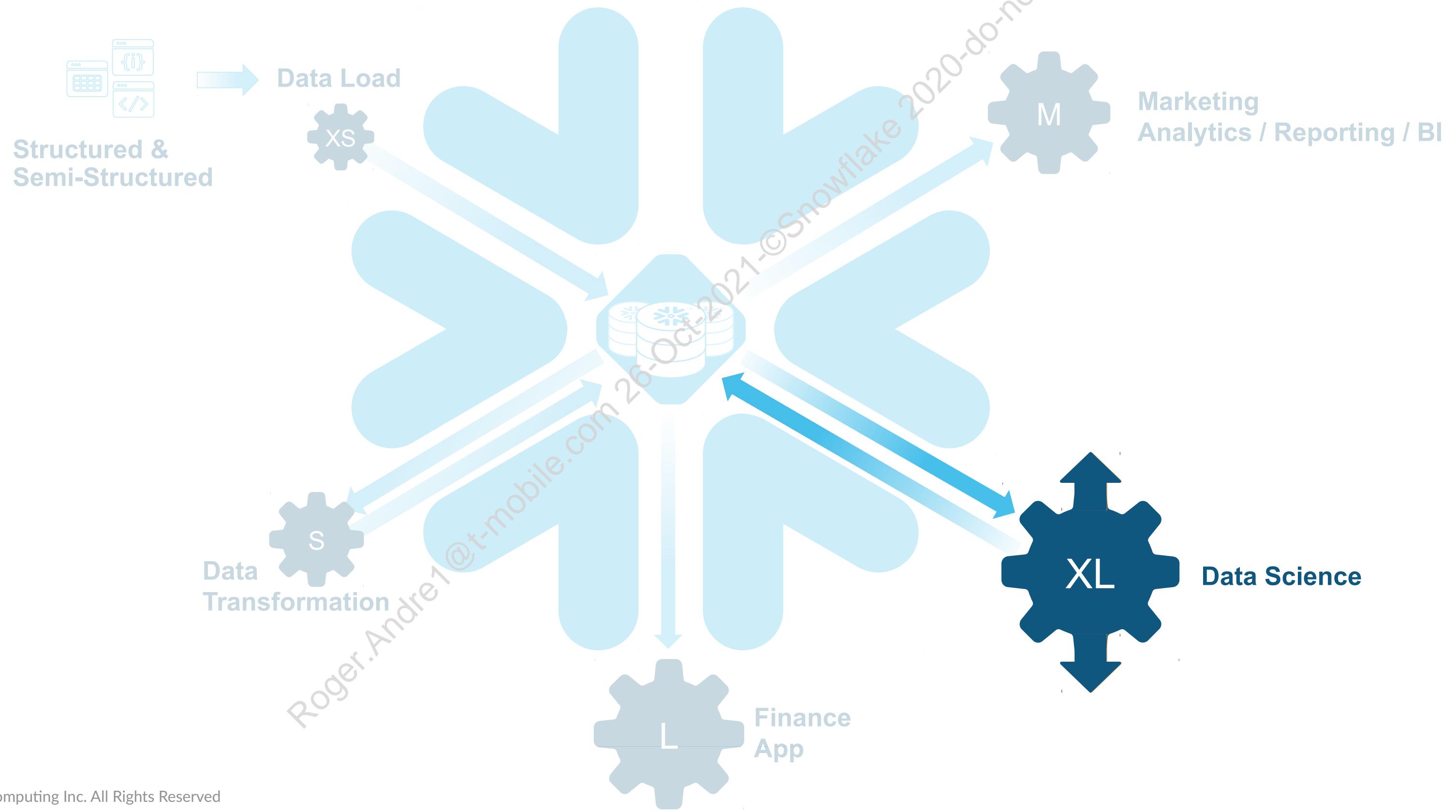
FUNCTIONAL ARCHITECTURE



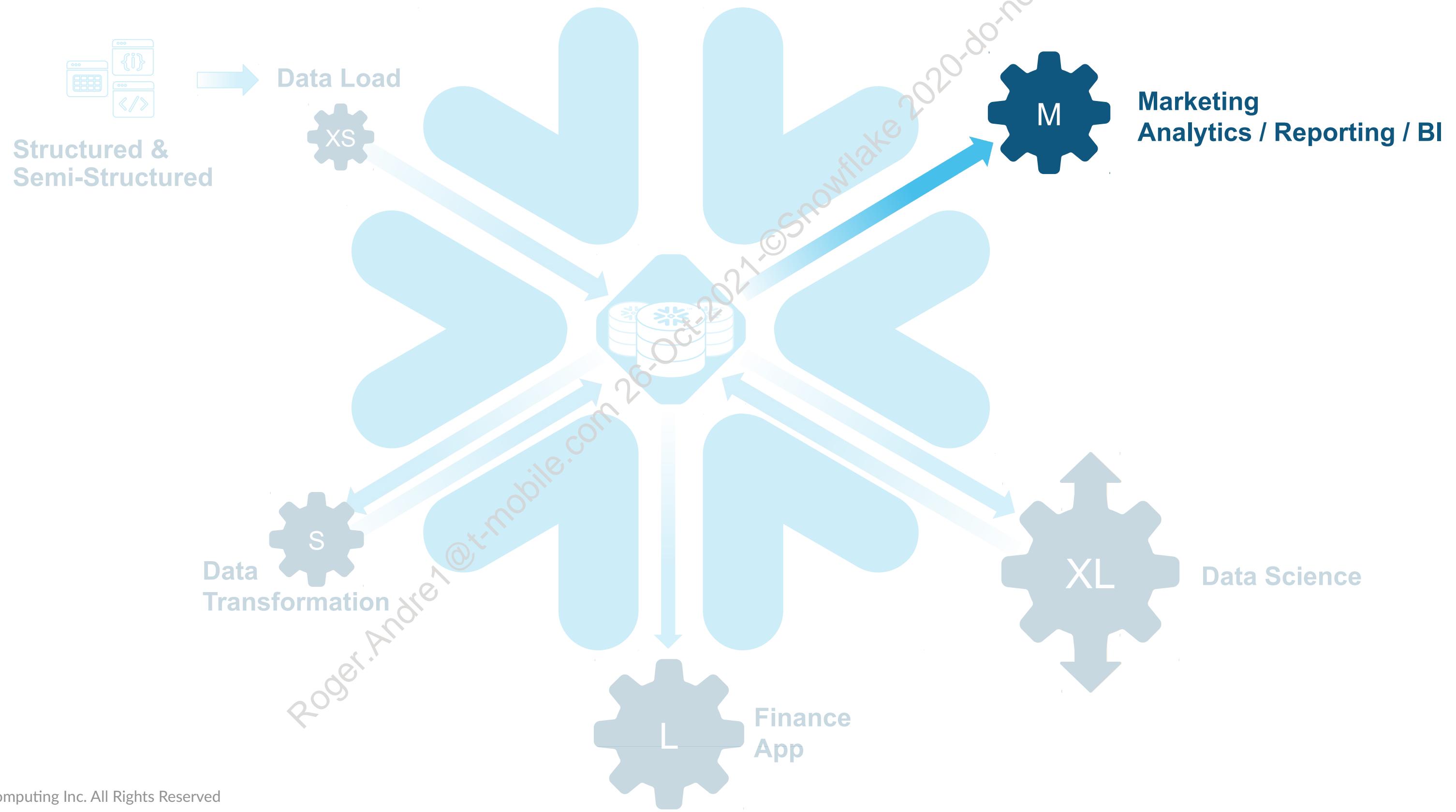
FUNCTIONAL ARCHITECTURE



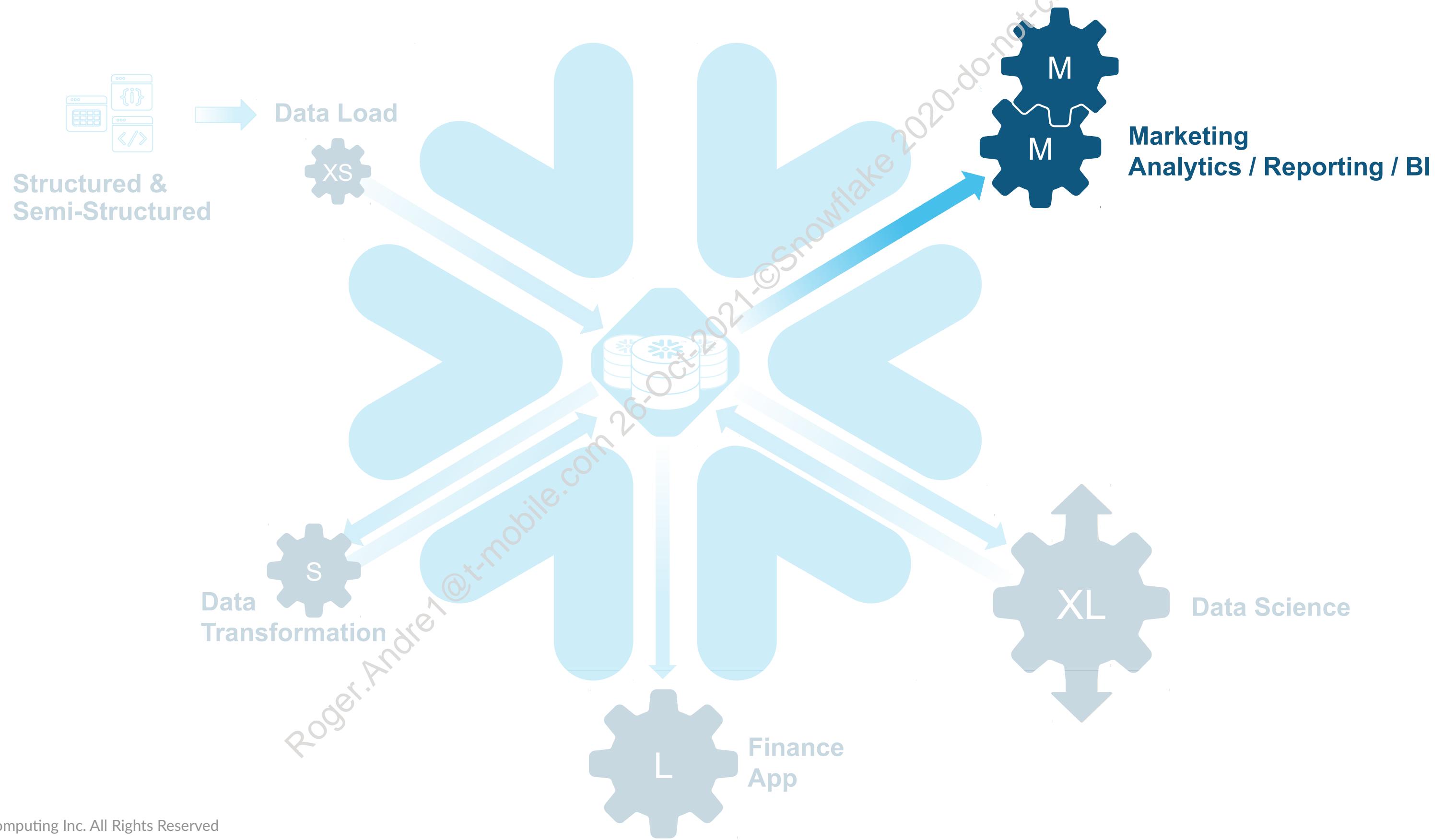
FUNCTIONAL ARCHITECTURE



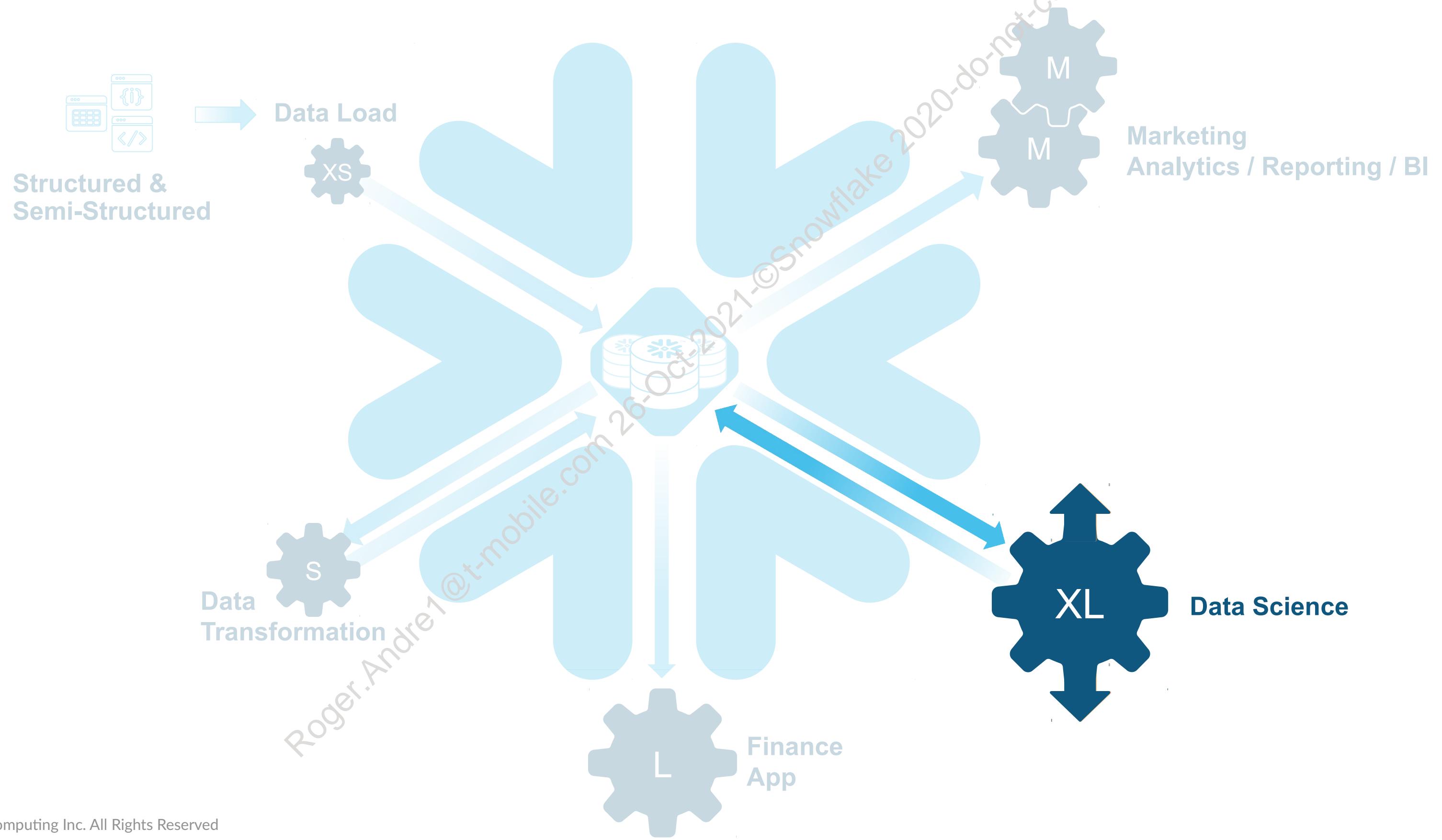
FUNCTIONAL ARCHITECTURE



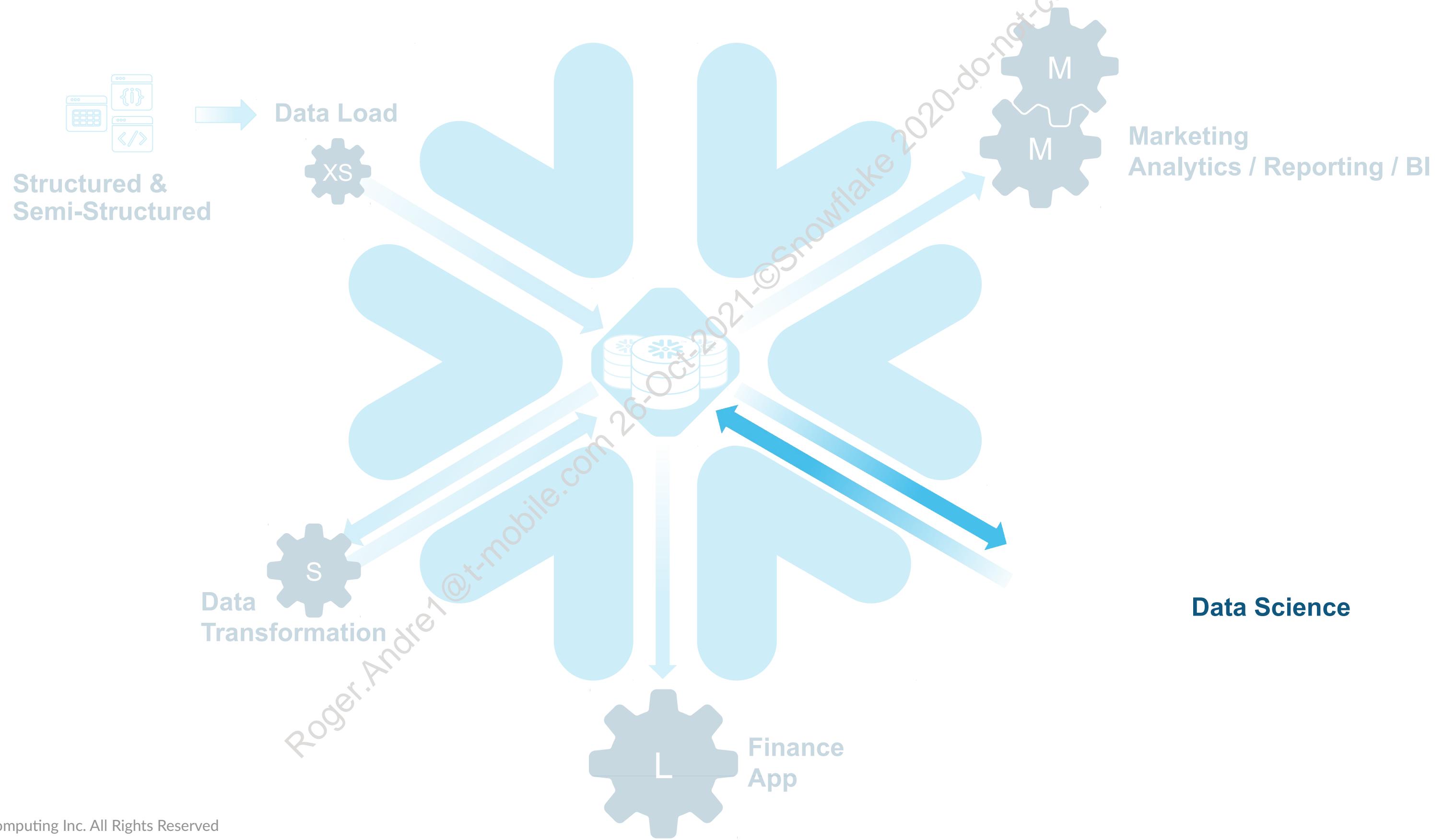
FUNCTIONAL ARCHITECTURE



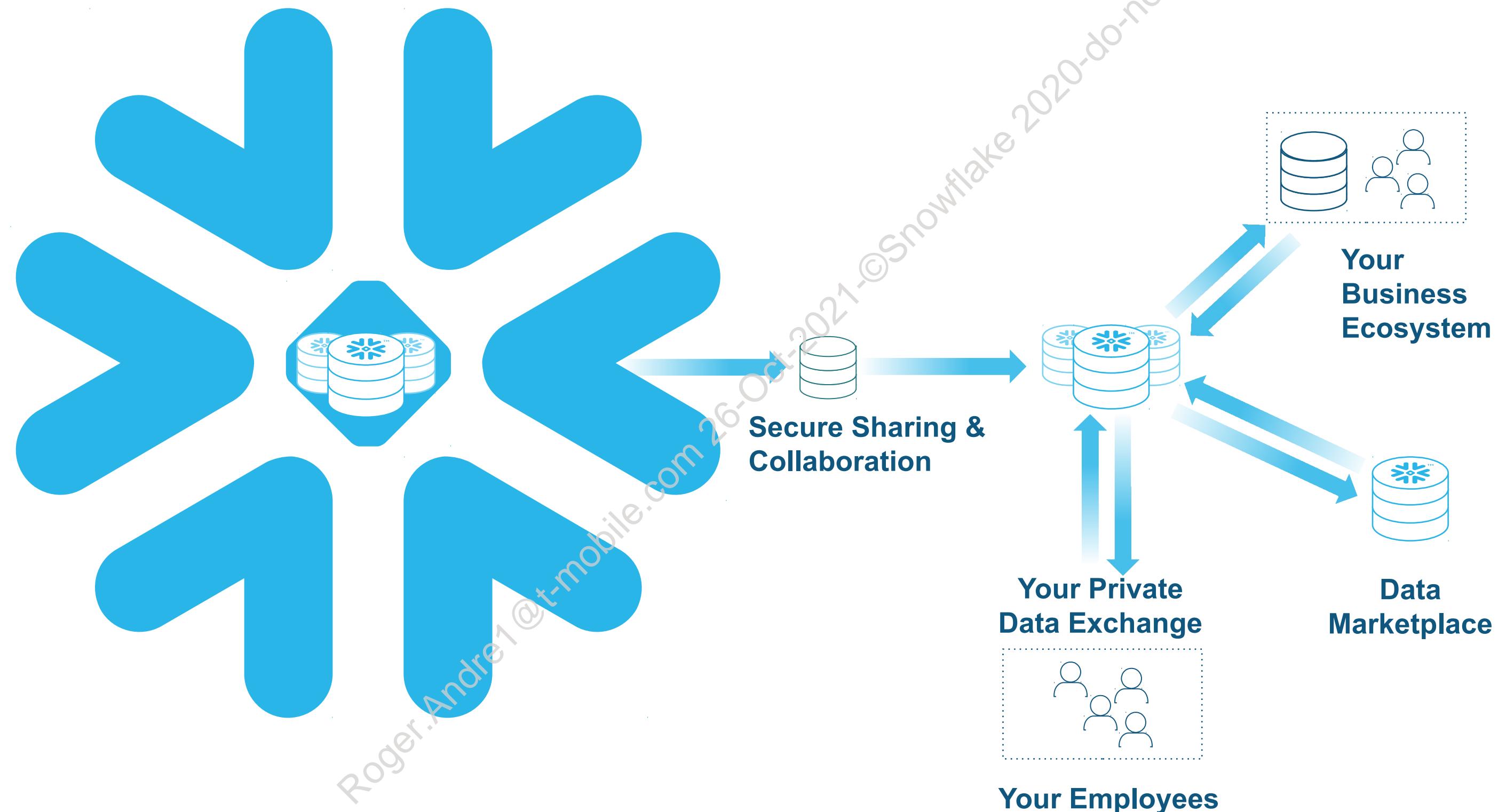
FUNCTIONAL ARCHITECTURE



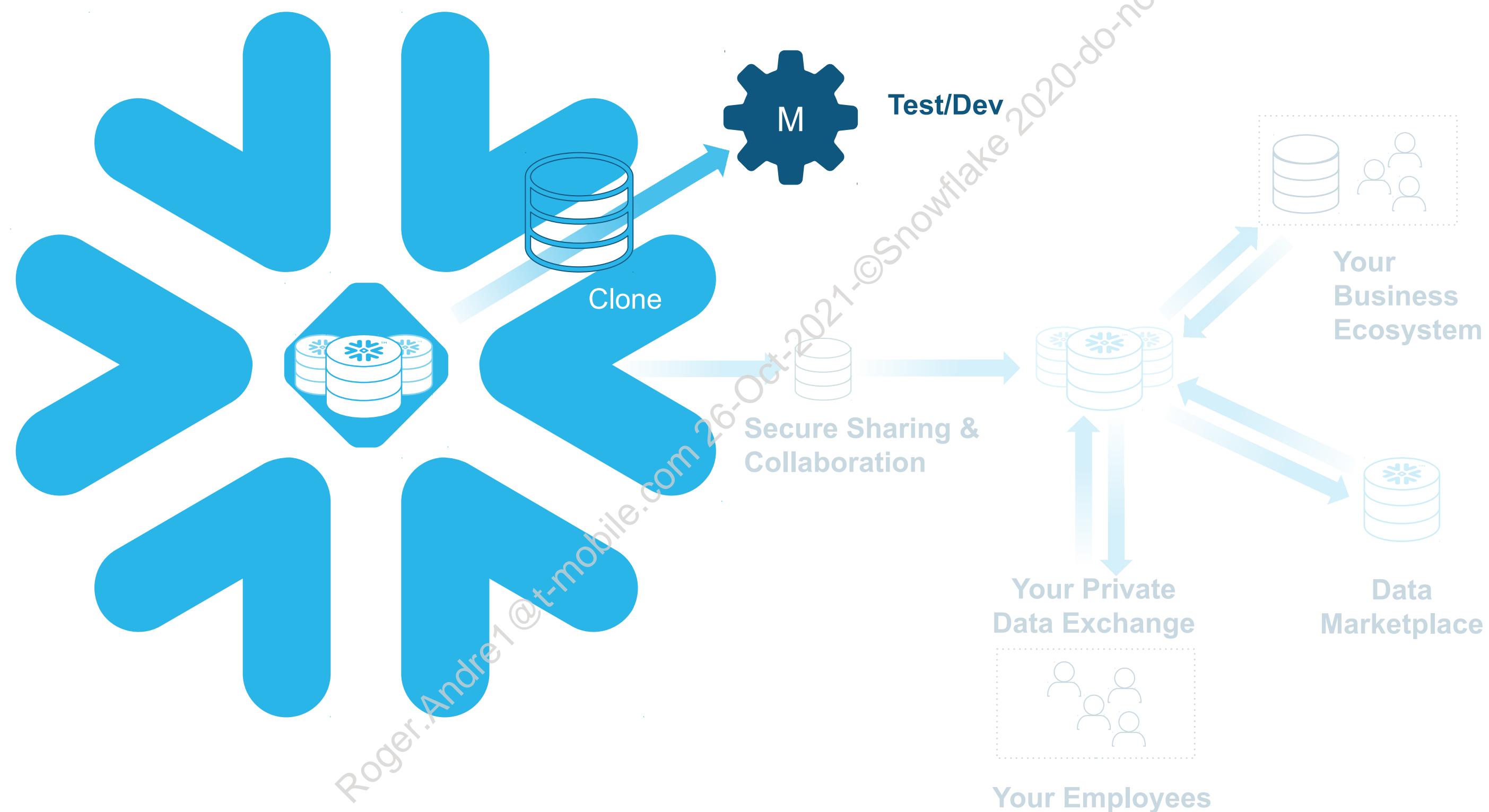
FUNCTIONAL ARCHITECTURE



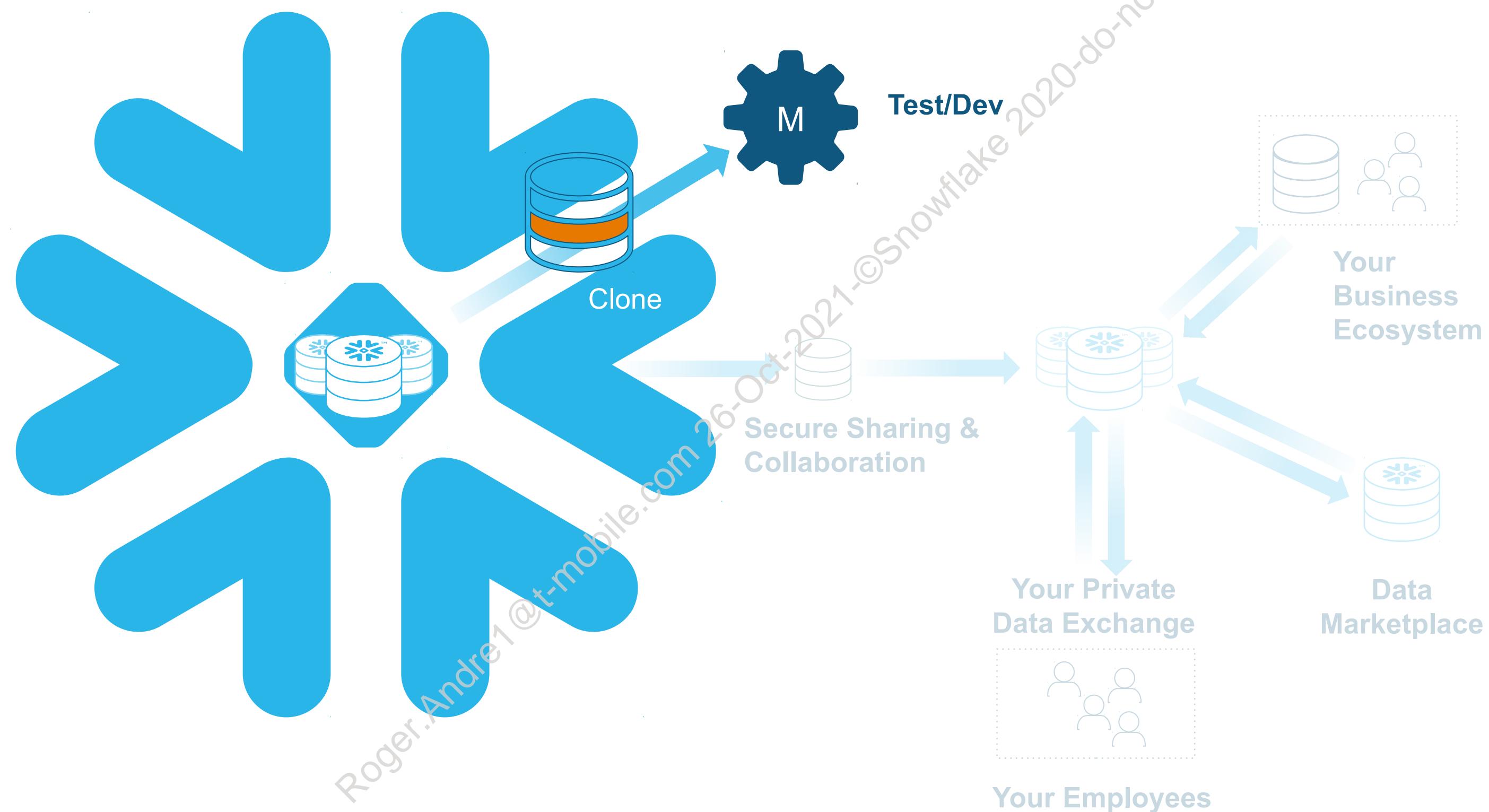
FUNCTIONAL ARCHITECTURE



FUNCTIONAL ARCHITECTURE



FUNCTIONAL ARCHITECTURE



SNOWFLAKE EDITIONS

STANDARD

Complete SQL data warehouse
Secure data sharing
Premier support 24x365
1 day of time travel
Enterprise-grade encryption
Dedicated virtual warehouses
Federated Authentication
Database Replication
External Functions
Snowsight
Create your own data exchange
Data Marketplace access

ENTERPRISE

Standard +
Multi-cluster warehouses
Up to 90 days of time travel
Annually rekey encrypted data
Materialized Views
Search Optimization Service
Dynamic Data Masking
External Data Tokenization

BUSINESS CRITICAL

Enterprise +
HIPAA support
PCI compliance
Data encryption everywhere
Tri-Secure secure
AWS PrivateLink support
Azure Private Link support
Database failover and fallback

VIRTUAL PRIVATE SNOWFLAKE (VPS)

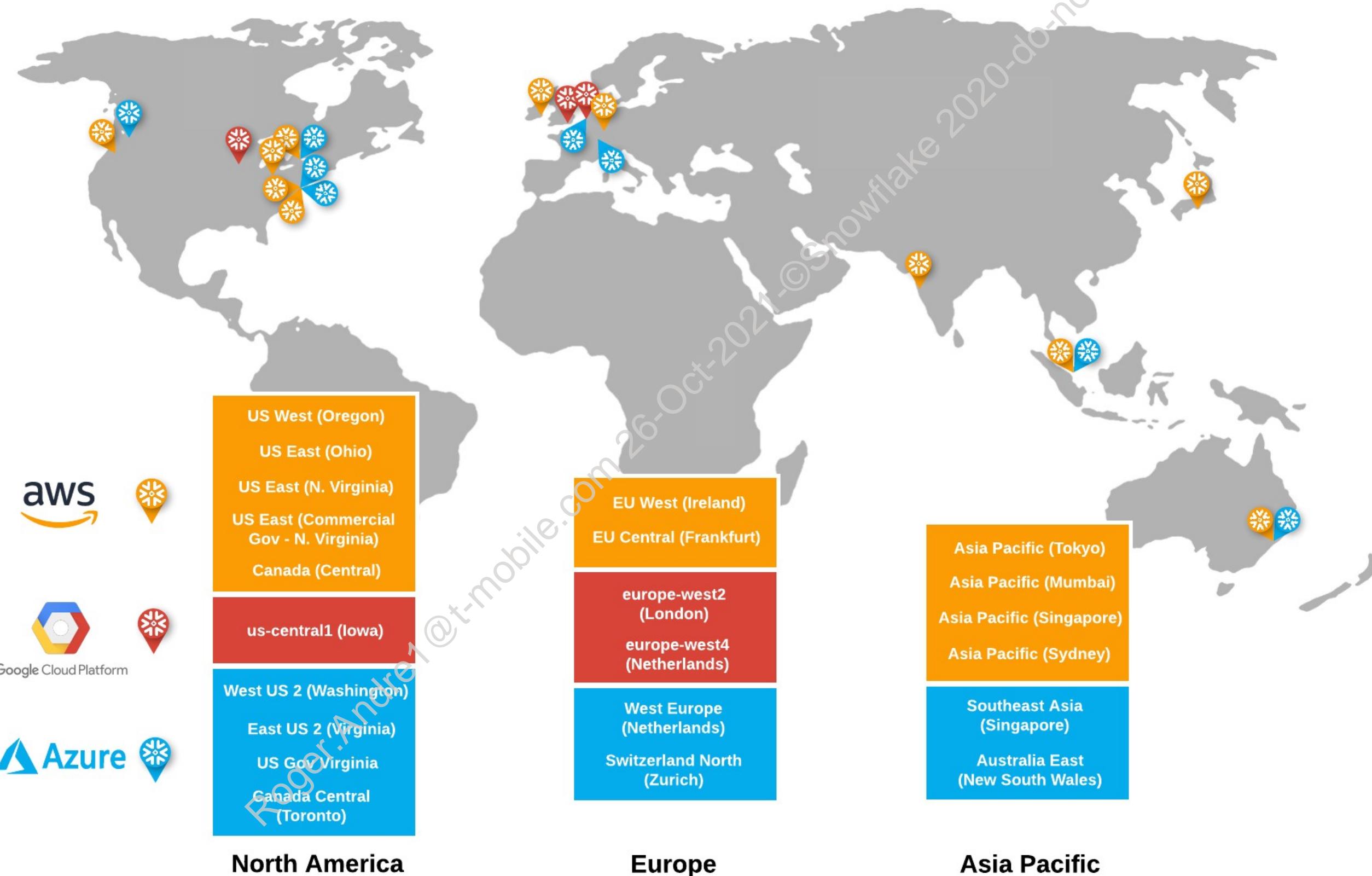
Business Critical +
Customer-dedicated virtual servers wherever the encryption key is in memory

Customer-dedicated metadata store



GLOBAL SNOWFLAKE

CURRENTLY SUPPORTED REGIONS



NAVIGATING THE SNOWFLAKE UI

Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy

INSTRUCTOR DEMO

Using the Snowflake Web UI

15 minutes

- Overview
- Worksheet Context
- Tips and Tricks



LAB EXERCISE: 1

Take a Quick Test Drive

25 minutes

Tasks:

- Log in to your Snowflake training account
- Create a database and a table for later use
- Run queries on sample data

Note:

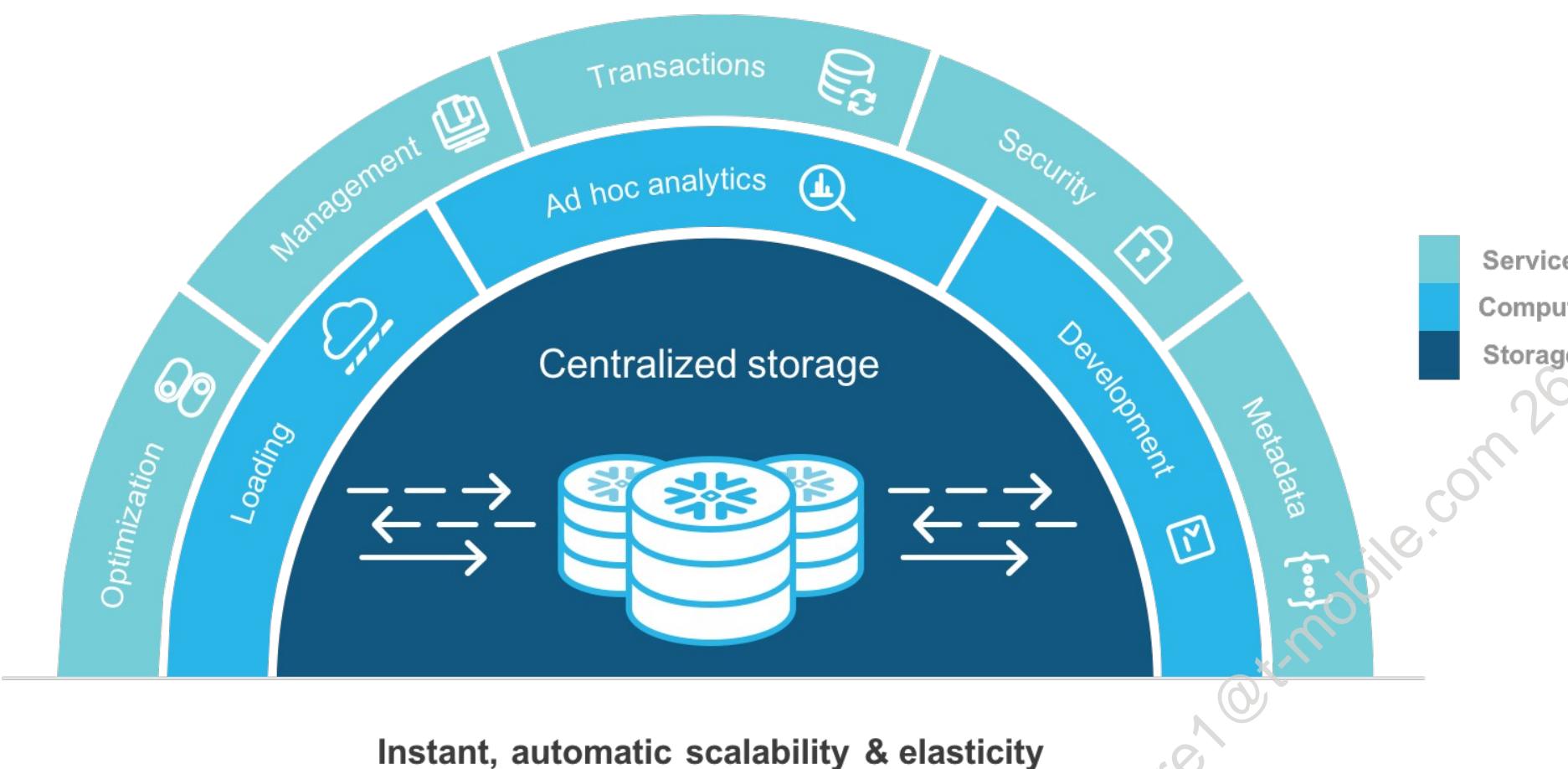
- In this lab, you will define your user's default role, namespace, and warehouse so they will automatically be set in every worksheet you open.
- Internet Explorer is not supported!



CLOUD SERVICES LAYER

Roger.Andre1@t-mobile.com 26-Oct-2021 © Snowflake 2020-do-not-copy

SNOWFLAKE ARCHITECTURE



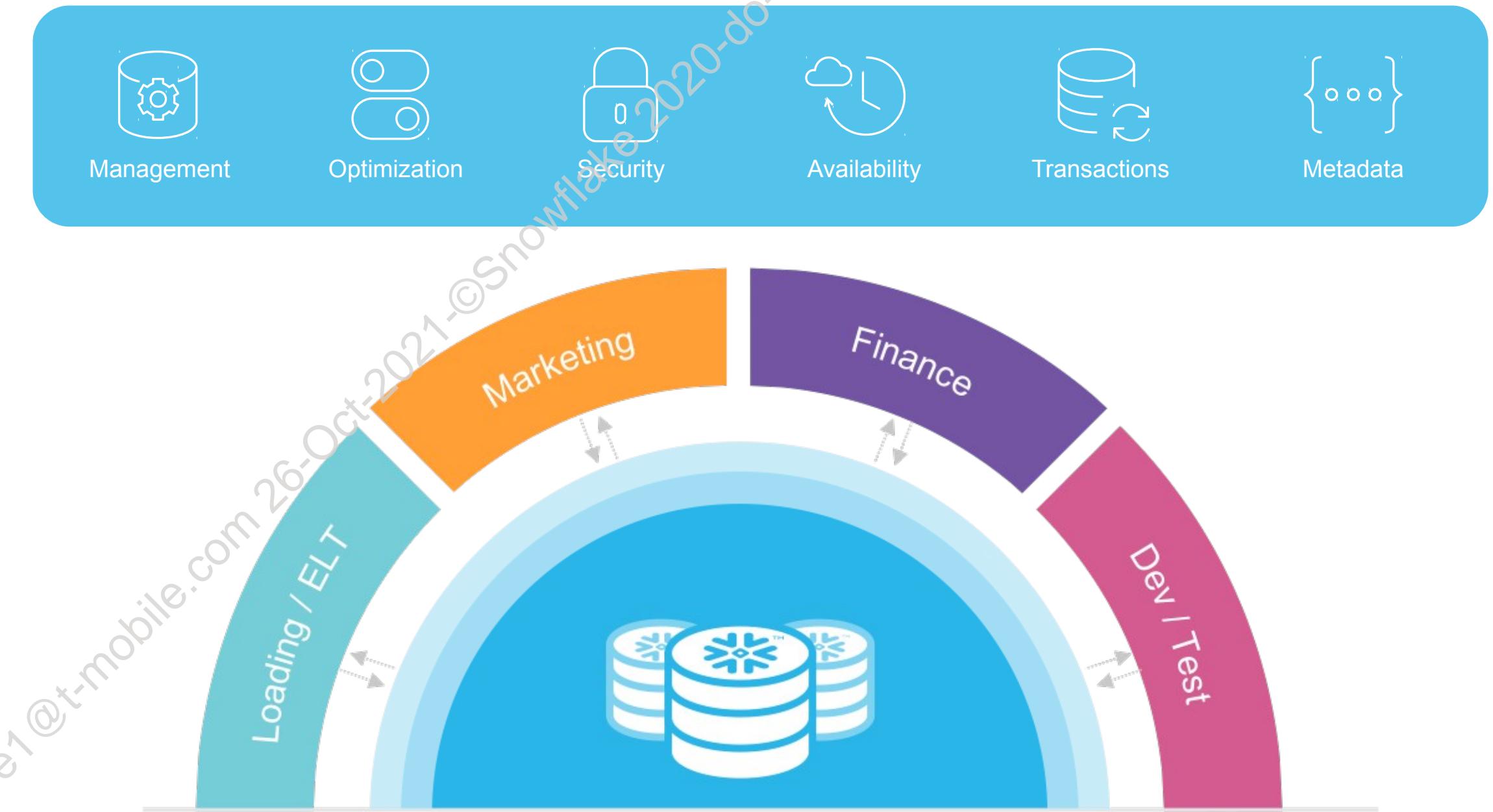
Three (3) Architectural Layers:

- Storage Layer
- Compute (Virtual Warehouse) Layer
- Cloud Services Layer

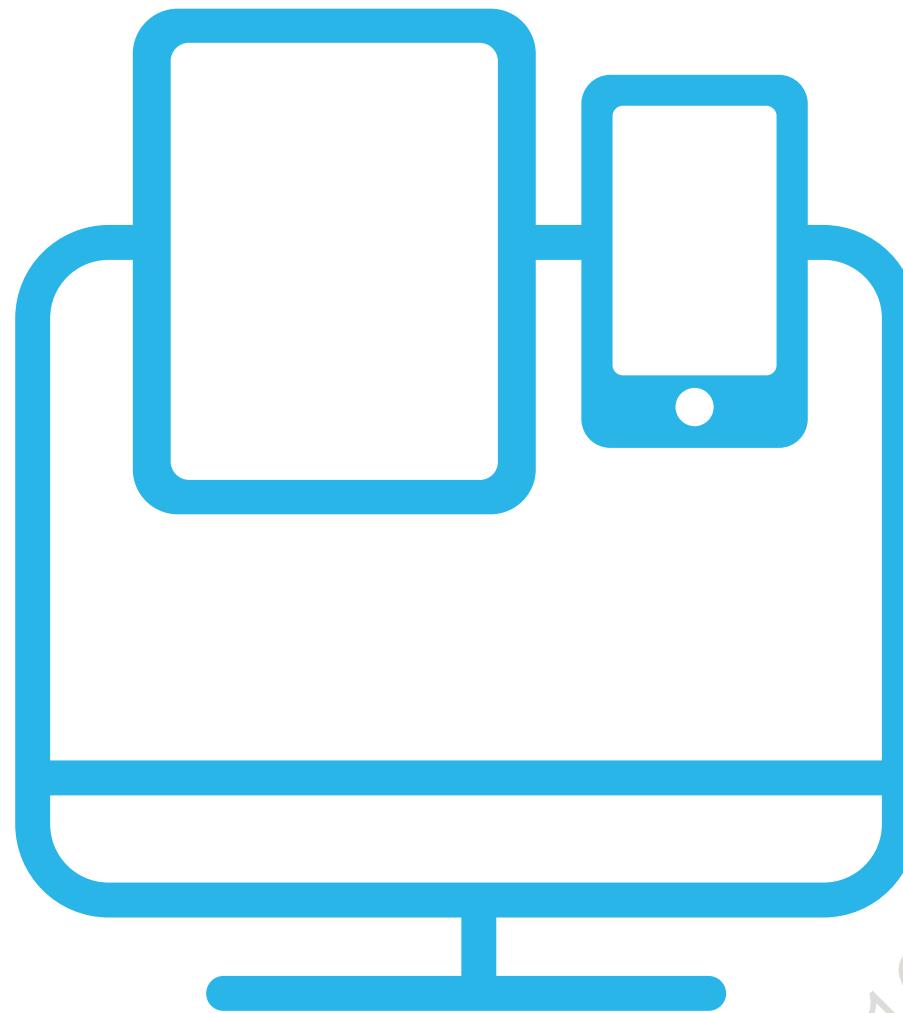


SNOWFLAKE ARCHITECTURE: CLOUD SERVICES

- “Brains” of the service
- Manages and coordinates activities across Snowflake
- Runs on compute instances provisioned by Snowflake

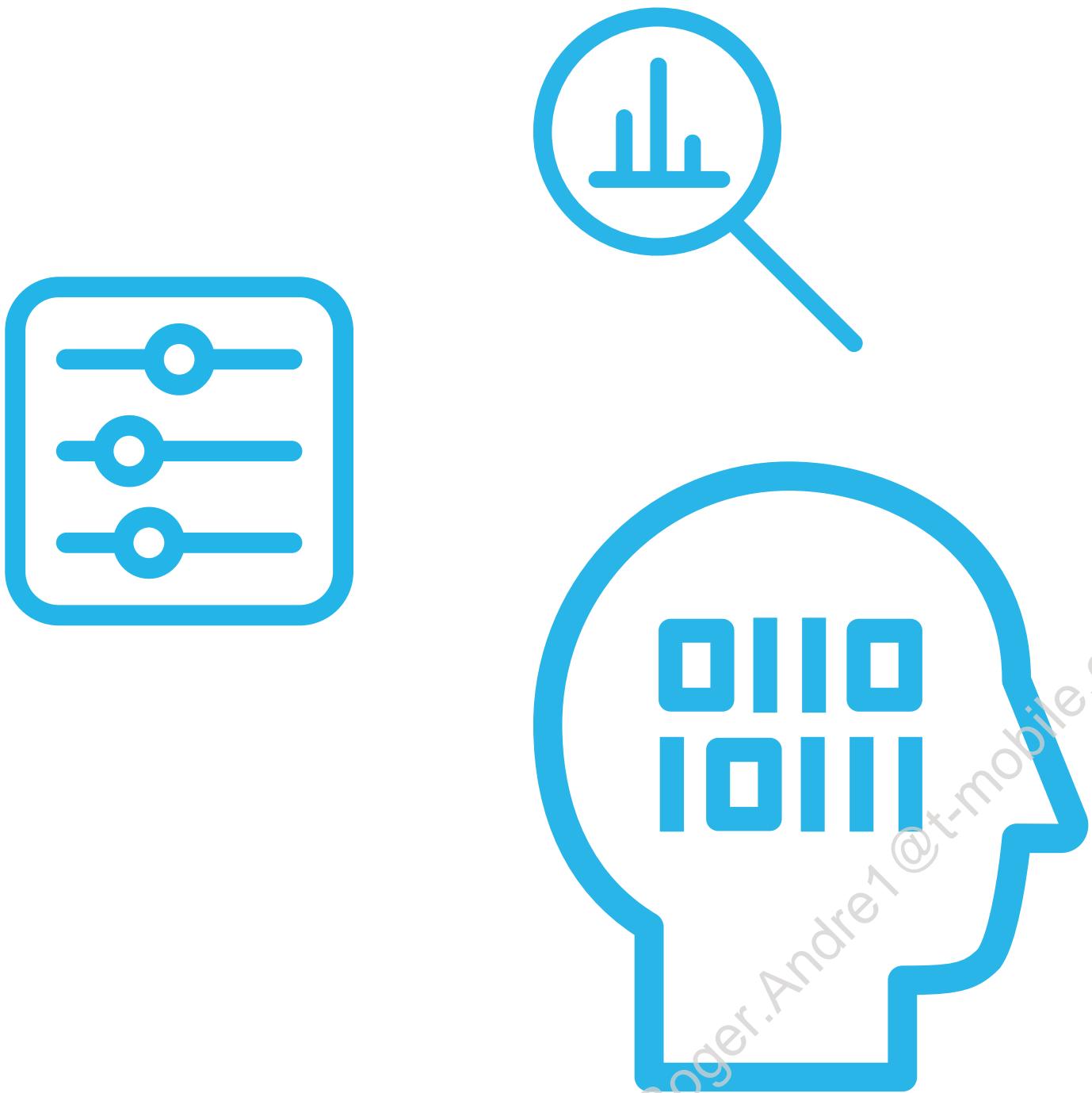


MANAGEMENT



- Centralized management for all storage
- Manages the compute that works with the storage
- Transparent, handling online updates and patches

OPTIMIZER SERVICE



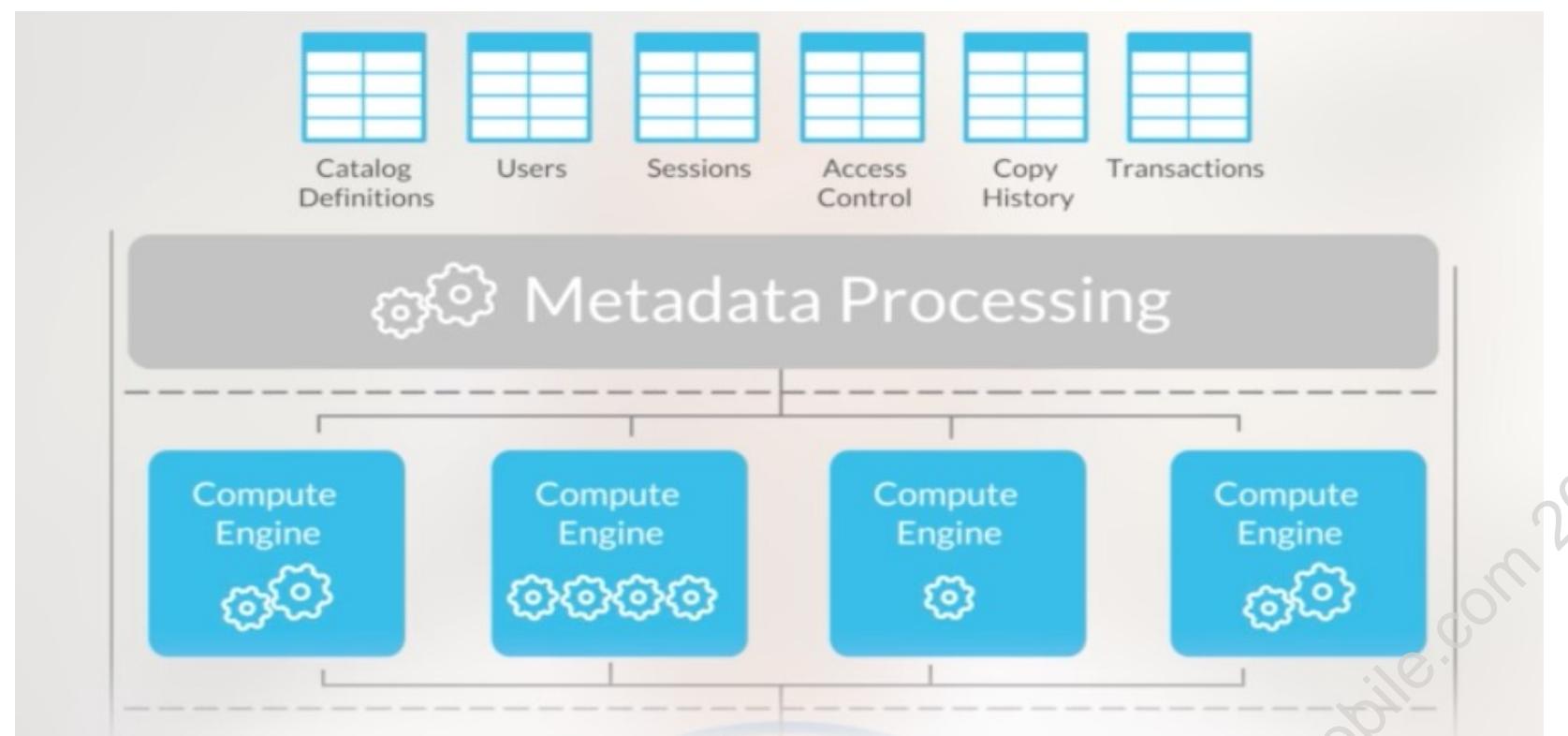
- SQL Optimizer
 - Cost-based optimization (CBO)
- Automatic JOIN order optimization
 - No user input or tuning required
- Automatic statistics gathering
- Pruning using metadata about micro-partitions

SECURITY



- Authentication
- Access control for users and roles
- Access control for shares
- Encryption and key management

METADATA MANAGEMENT



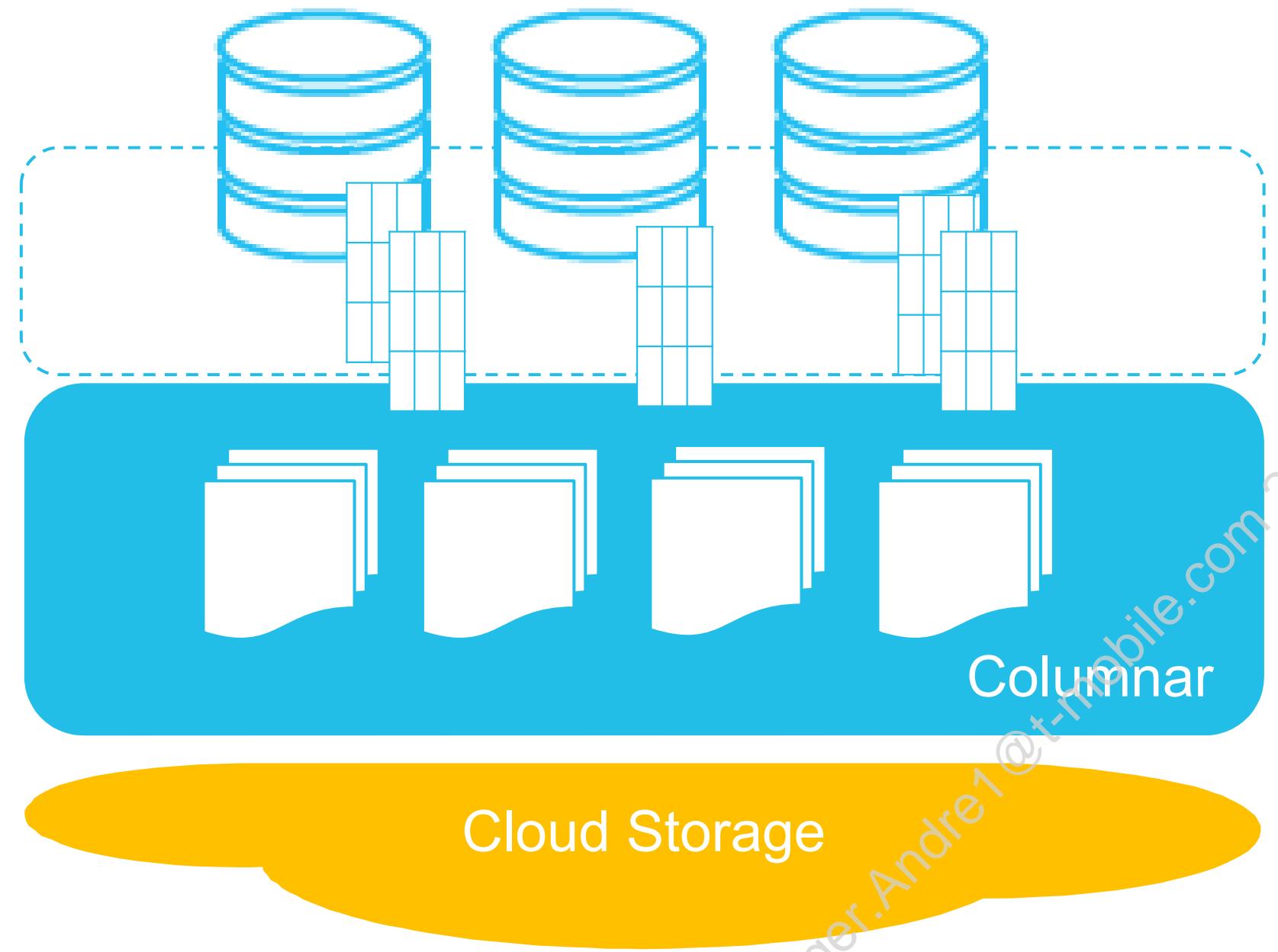
- Stores metadata as data is loaded into the system
- Handles queries that can be processed completely from metadata
- Used for Time Travel and Cloning
- Every aspect of Snowflake architecture leverages metadata



DATA STORAGE LAYER

Roger.Andre1@t-mobile.com 26-Oct-2021 © snowflake 2020-do-not-copy

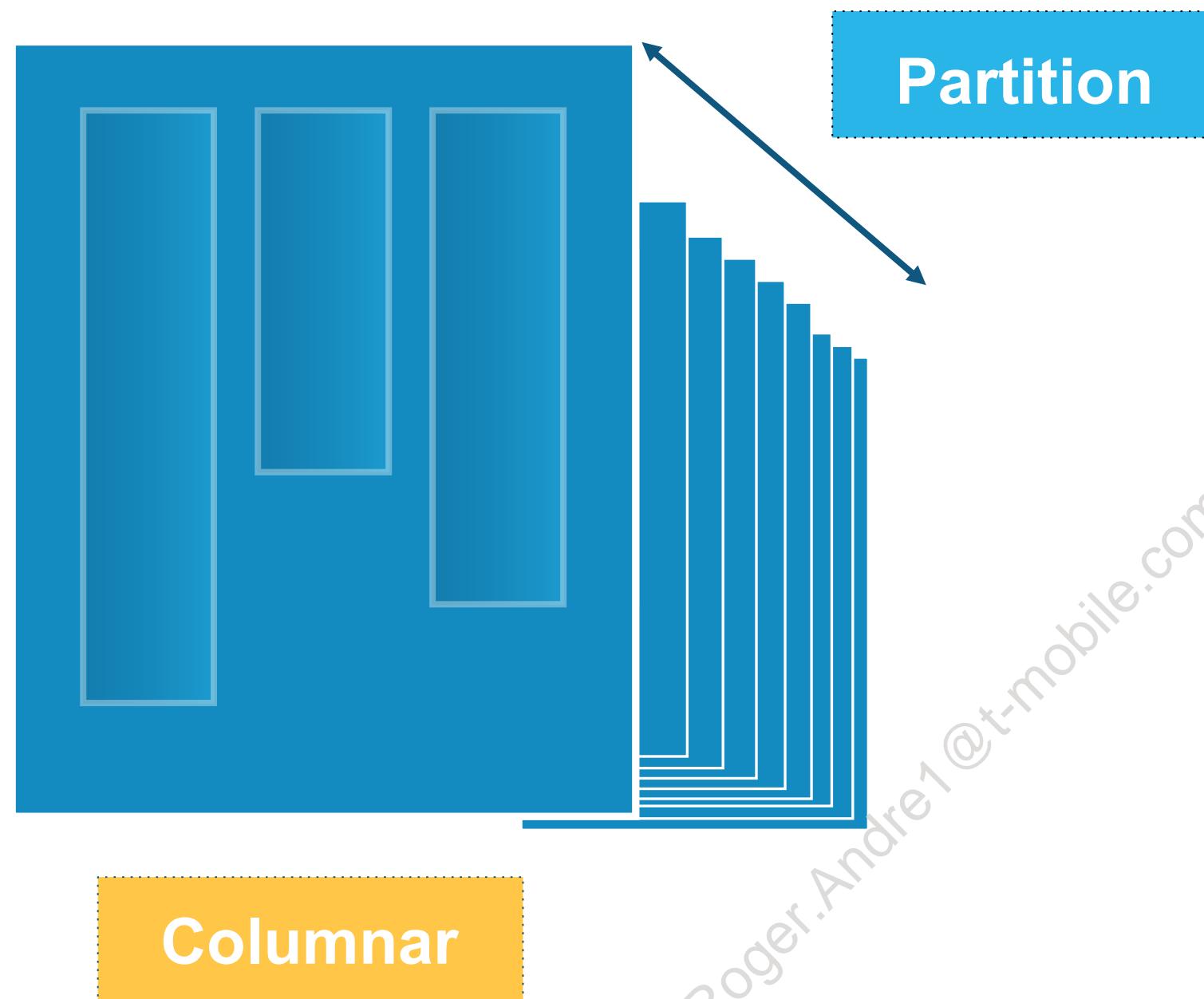
STORAGE LAYER



- Hybrid Columnar
- Automatic micro-partitioning
- Natural data clustering and optimization
- Native Semi-Structured data support and optimization
- All storage within Snowflake is billable (compressed)



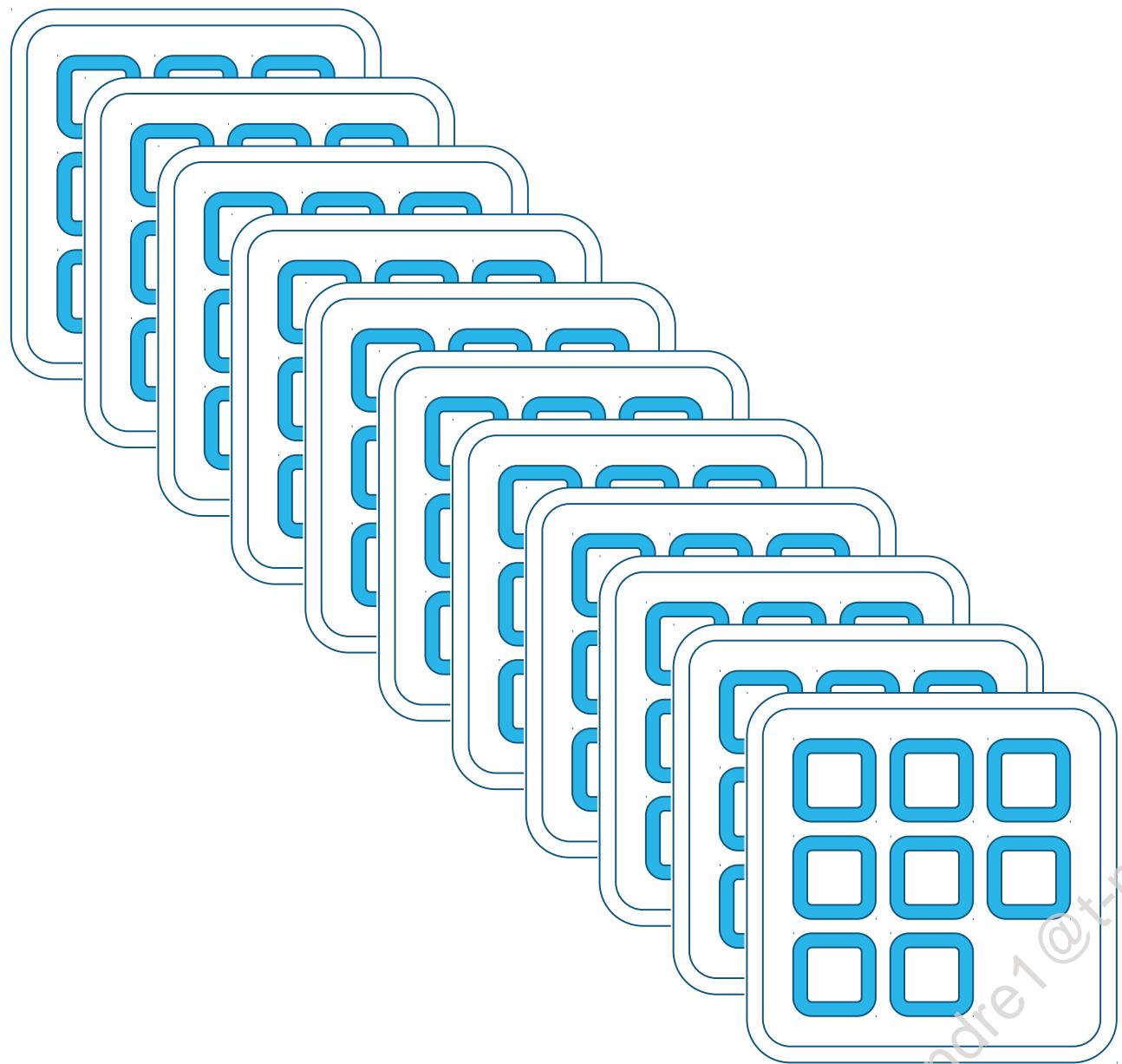
COLUMNAR COMPRESSION



- Ingestion automatically analyzes and compresses data into table on load
- Finds the optimal compression scheme for each data type
- Columns grow and shrink independently
- Significant performance benefit by reducing I/O and storage



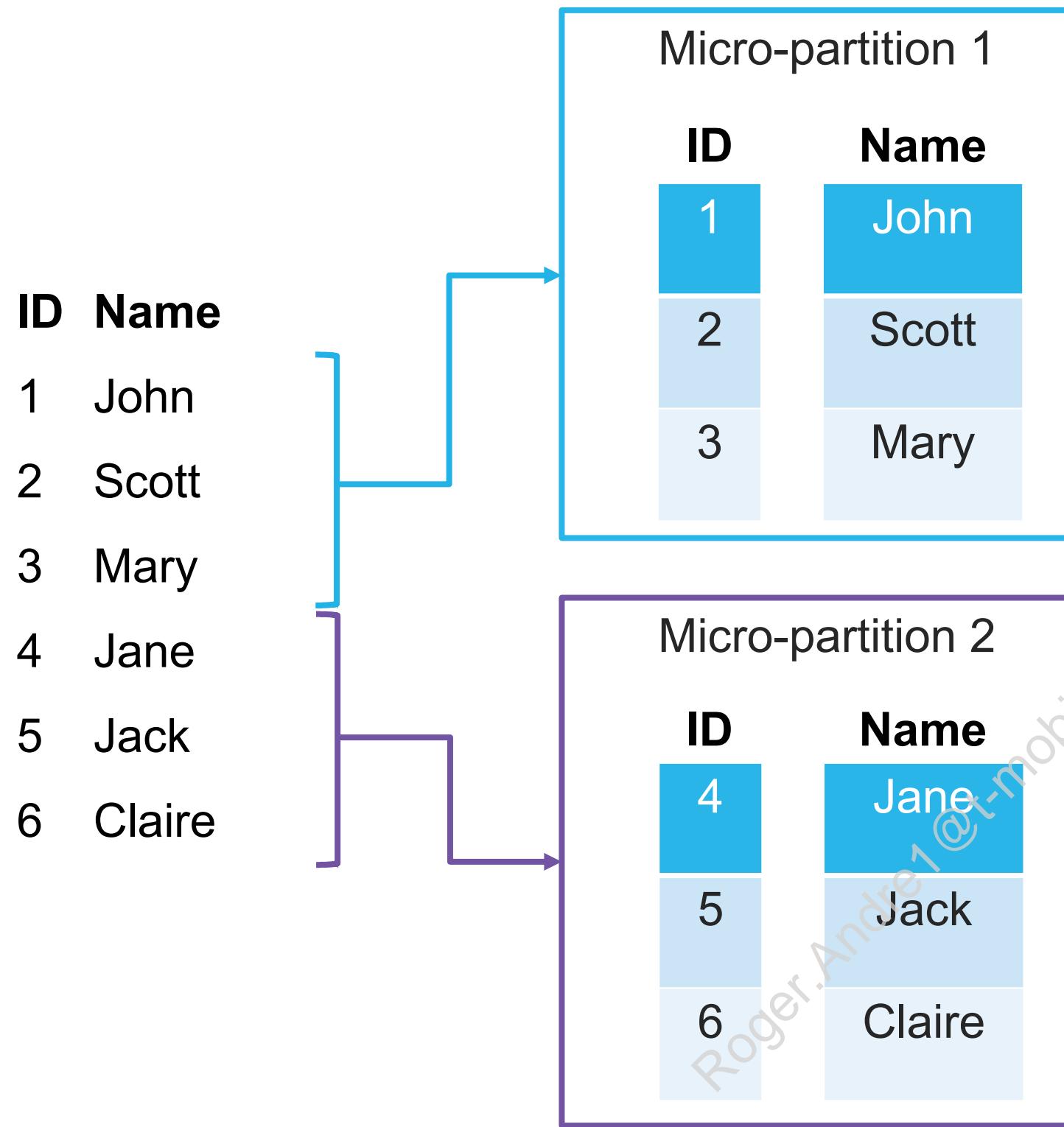
MICRO-PARTITIONS



- Contiguous units of storage that hold table data
 - 50 - 500 MB of uncompressed data
 - Generally...Max 16MB (Compressed)
- MANY micro-partitions per table
- IMMUTABLE !!!!!
- Services layer stores metadata about every micro-partition
 - MIN/MAX (Range of values in each column)
 - Number of distinct values



MICRO-PARTITIONING



- Physical data files that comprise Snowflake's logical tables
- Automatically-created contiguous units of storage, partitioned based on ingestion order
- Attempts to preserve natural data co-location
- Immutable - updates create new micro-partition versions



METADATA

Micro-partition Files

ID	Name
1	John
2	Scott
3	Mary

Micro-partition Metadata

ID: 1 - 3
Name: J - S

ID	Name
4	Jane
5	Jack
6	Claire

ID: 4 - 6
Name: C - J

- Snowflake automatically collects and maintains metadata about tables and their underlying micro-partitions, including:

- Table level
 - Row count
 - Table size (in bytes)
 - File references and table versions

- Micro-partition column level:

- Range of values
 - Number of distinct values
 - MIN and MAX values
 - NULL count



DATA STORAGE BILLING

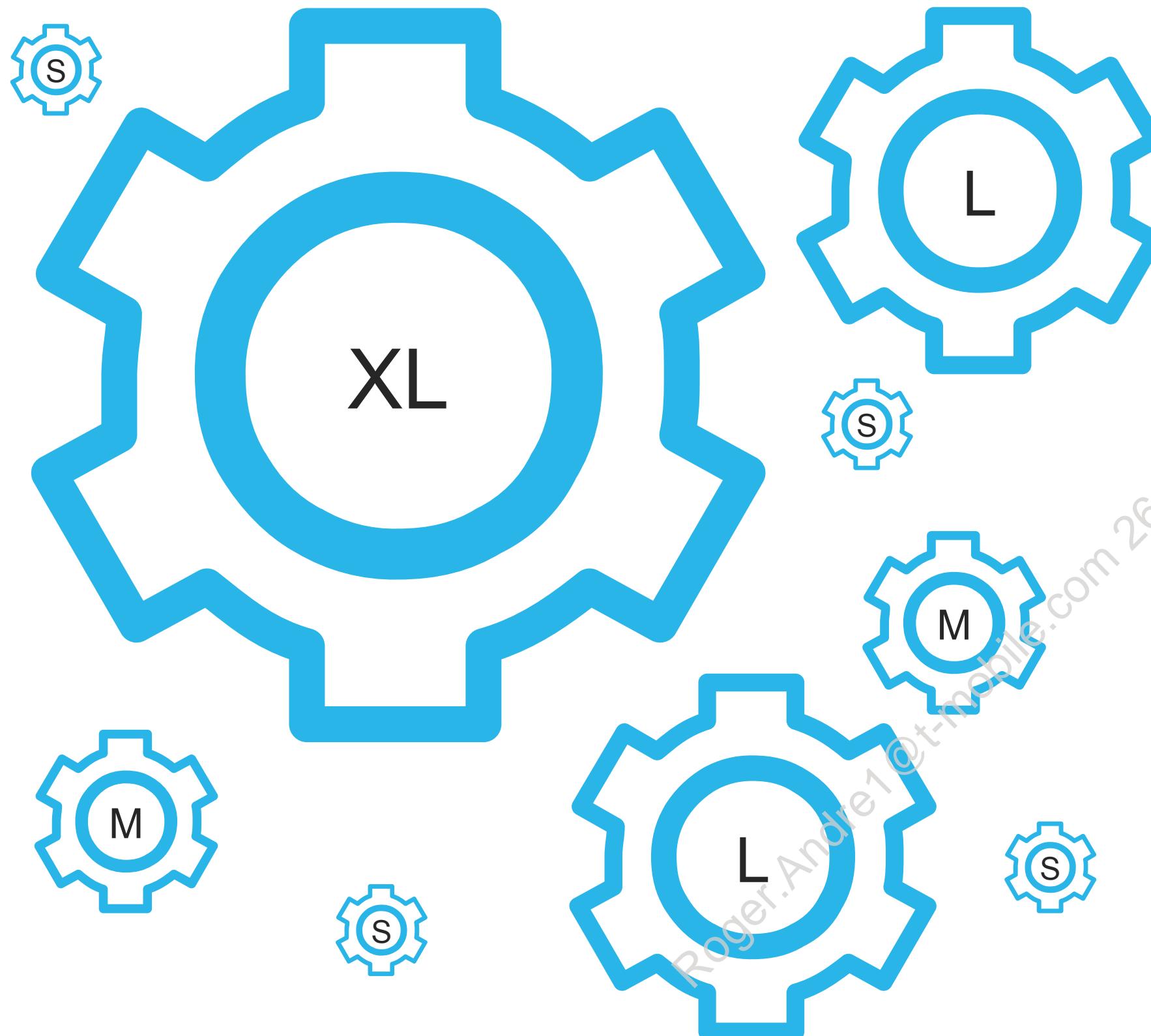
- Billed for actual storage use
 - Daily average Terabytes per month
- On-demand pricing
 - Billed in arrears for storage used
 - Around \$40/Terabyte/month
 - Minimum monthly charge of \$25
- Pre-Purchased Capacity
 - Billed up front with commitment to a certain capacity
 - Price varies on amount and cloud platform
 - Customer is notified at 70% of capacity



COMPUTE LAYER

Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy

VIRTUAL WAREHOUSES



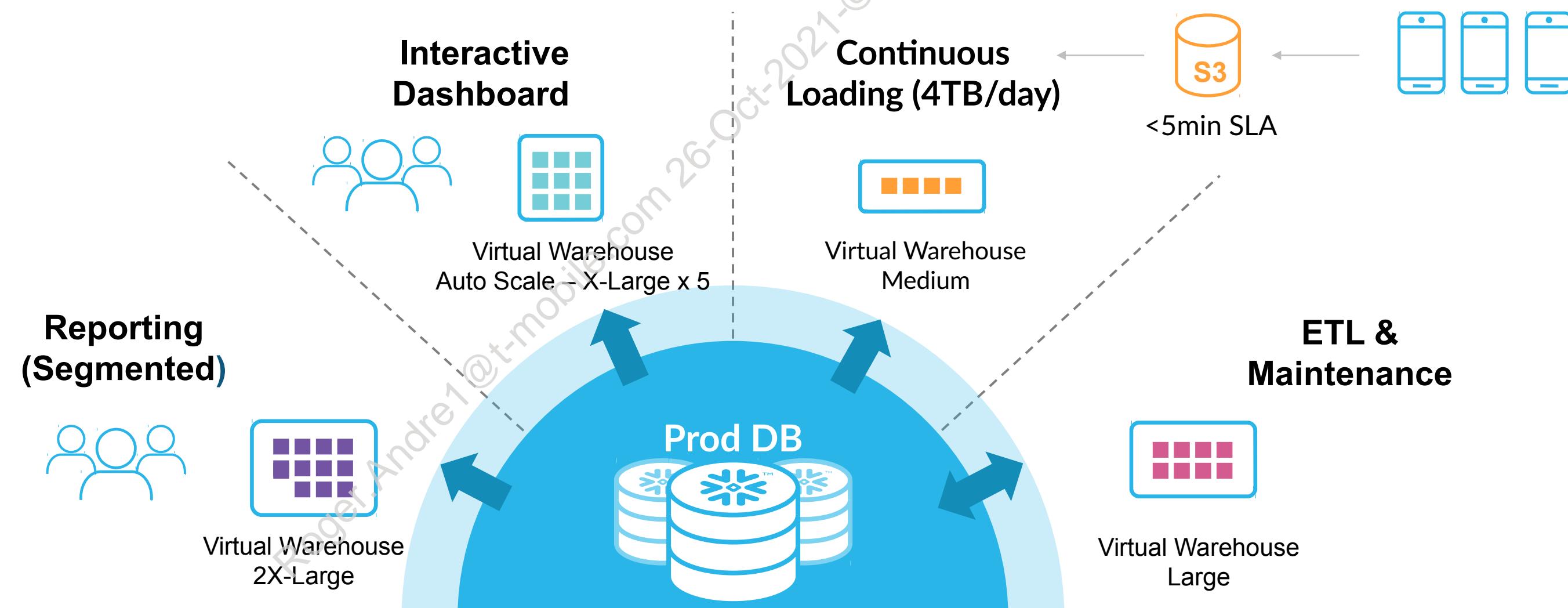
- A named wrapper around a cluster of servers with CPU, memory, and disk
- The larger the warehouse, the more servers in the cluster
- Extra Small has one server per cluster
 - Each size up doubles in size
- Jobs requiring compute run on virtual warehouses
- While running a virtual warehouse consumes Snowflake credits
 - You are charged for compute



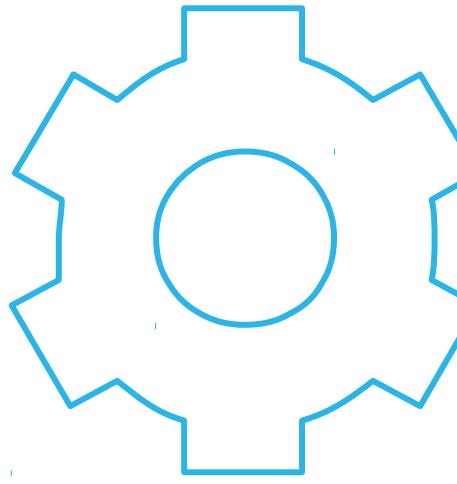
WORKLOAD SEGMENTATION

- Should reflect units of workload management

- ETL
 - BI / Dashboards
 - Data Science

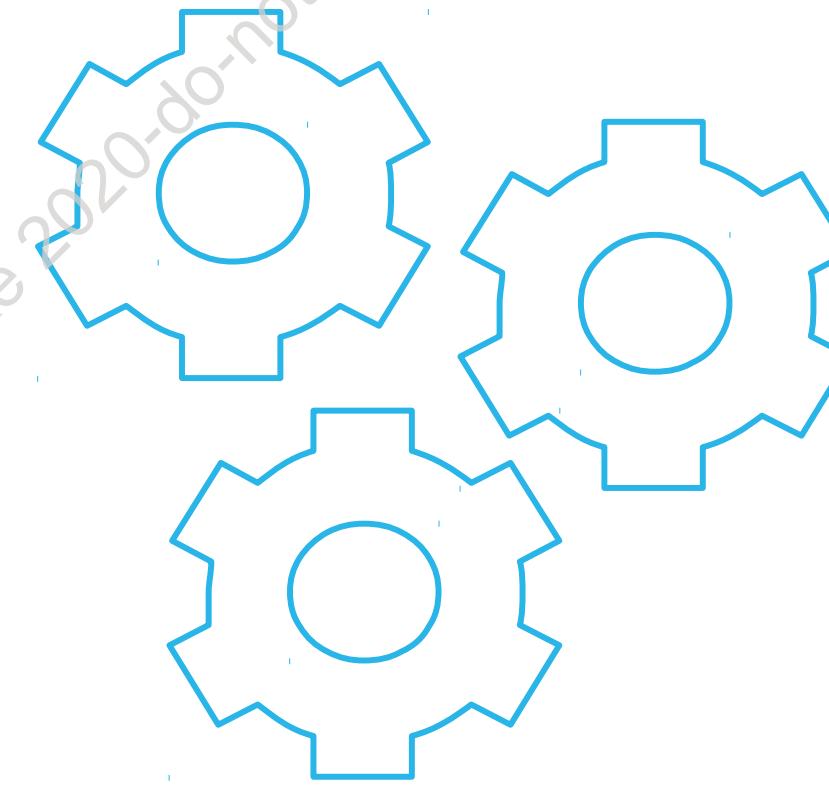


VIRTUAL WAREHOUSE TYPES



Standard

- Will only ever have a single compute cluster
- Cannot “scale out”

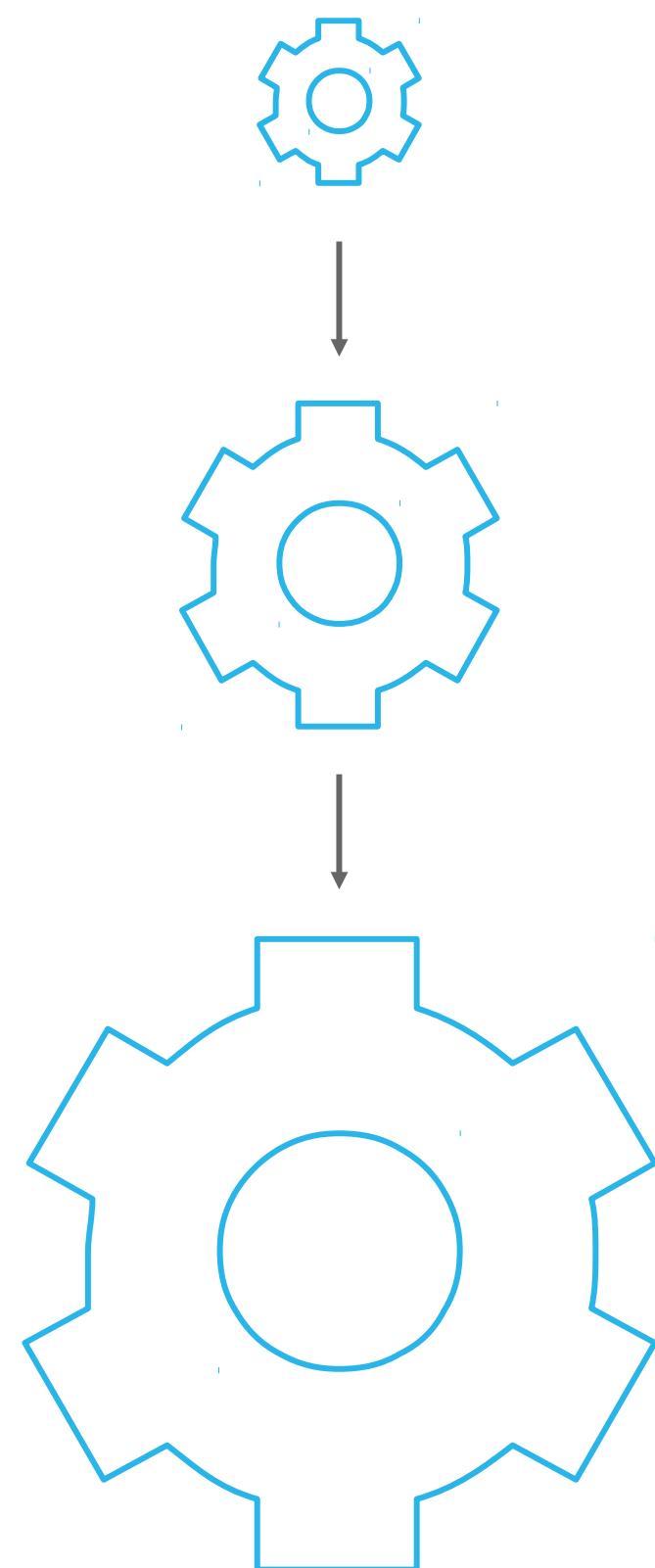


Multi-Cluster Warehouse (MCW)

- Can spawn additional compute clusters (scale out) to manage changes in user and concurrency needs
- Enterprise Edition feature

VIRTUAL WAREHOUSE SIZING

- Warehouses are sized in “t-shirt” sizing
- Size determines the number of servers that comprise each cluster in a warehouse
- Each larger size is double the preceding, in both VMs in the cluster and in Snowflake credits consumed



COMPUTE CREDITS



- How you are billed for compute usage
 - You may have a set number of credits
 - You may be billed monthly for your credits
- Charge based on the number of virtual warehouses you use, their size, and how long they are running
- Warehouse usage (or compute) is charged per-second, with a one-minute minimum



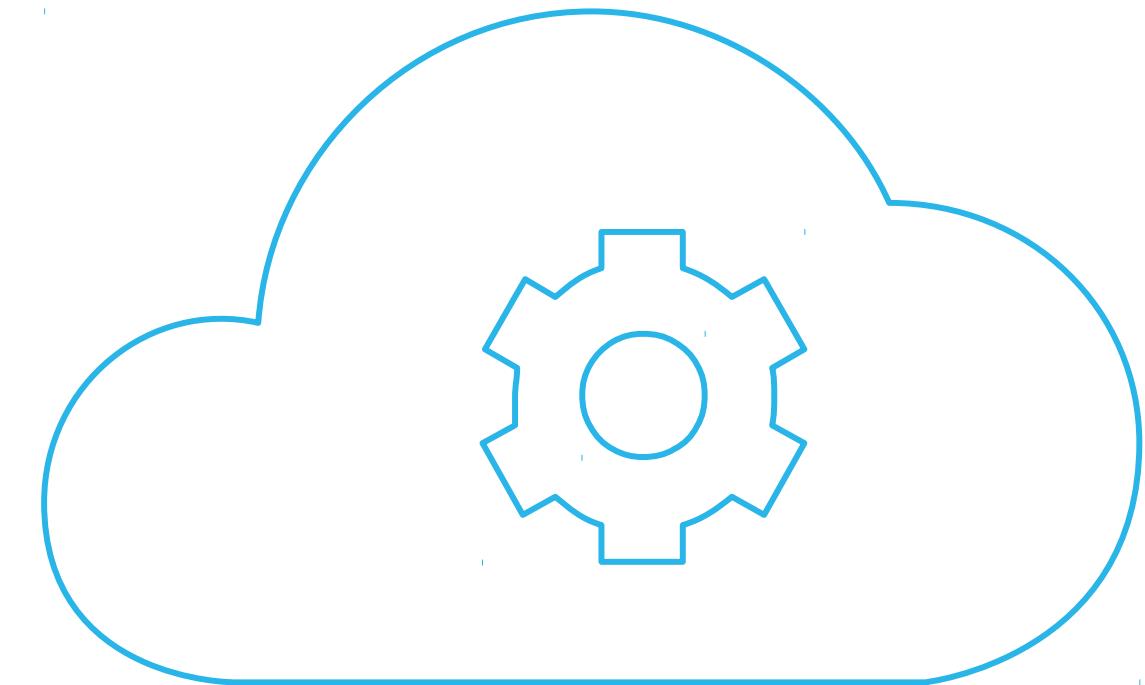
VIRTUAL WAREHOUSE CREDITS

Warehouse Size	Servers	Credits / Hour	Credits / Second
X-Small	1	1	0.0003
Small	2	2	0.0006
Medium	4	4	0.0011
Large	8	8	0.0022
X-Large	16	16	0.0044
2X-Large	32	32	0.0089
3X-Large	64	64	0.0178
4X-Large	128	128	0.0356



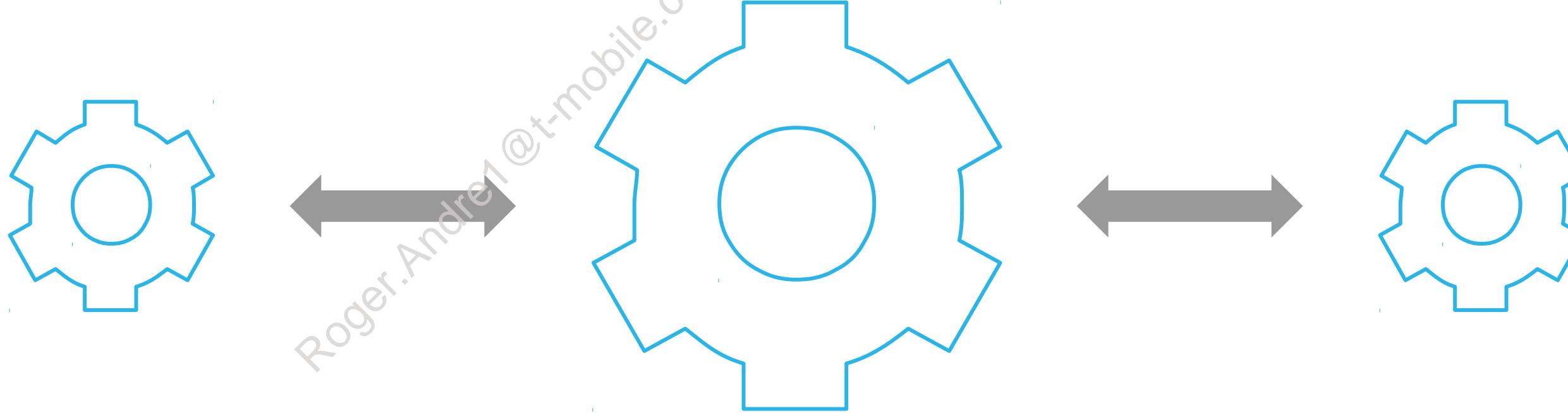
CLOUD SERVICES COMPUTE BILLING

- Some compute occurs in the cloud services layer
- Customers are charged for cloud computing that exceeds 10% of total compute costs for the account
- The METERING_HISTORY view inside the SNOWFLAKE.ACOUNT_USAGE schema will provide the number of credits used in the account. This information is also available in the WebUI.



RESIZING A WAREHOUSE

- Can be completed at any time, even when running
- Completed via `ALTER WAREHOUSE` statement or the UI
- Effects of resizing:
 - Suspended Warehouse: will start at new size upon next resume
 - Running Warehouse: immediate impact; running queries complete at current size, while queued queries run at new size



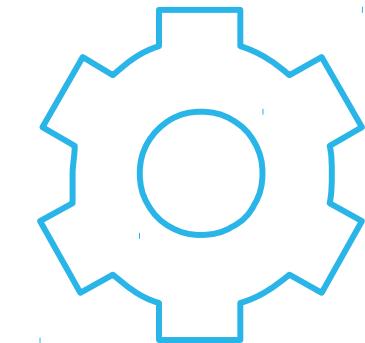
SCALE UP FOR PERFORMANCE

Elastic Processing Power (CPU, RAM, SSD)

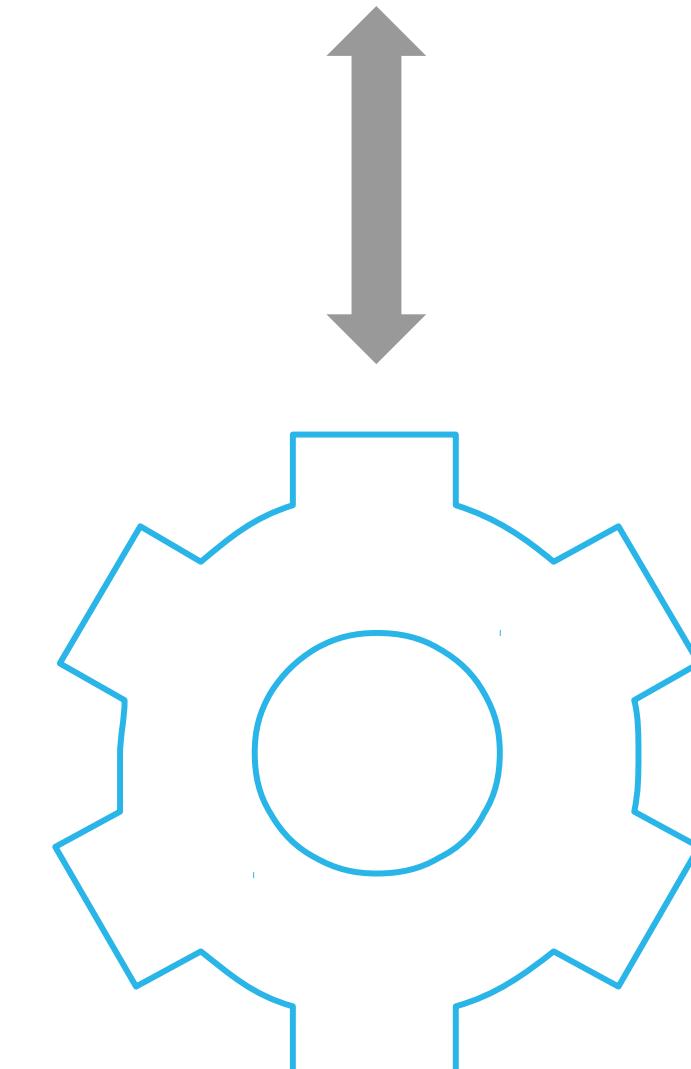
- Raw performance boost for complex queries or large data sets
- More complex queries on larger datasets require larger warehouses
- Not intended for handling concurrency issues (more users/queries)

Warehouse Sizing Guidelines

- Snowflake uses per-second billing: use larger warehouses for more complex workloads and (auto) suspend when not in use
- Keep queries of similar size and complexity on the same warehouse to simplify compute resource sizing



Medium



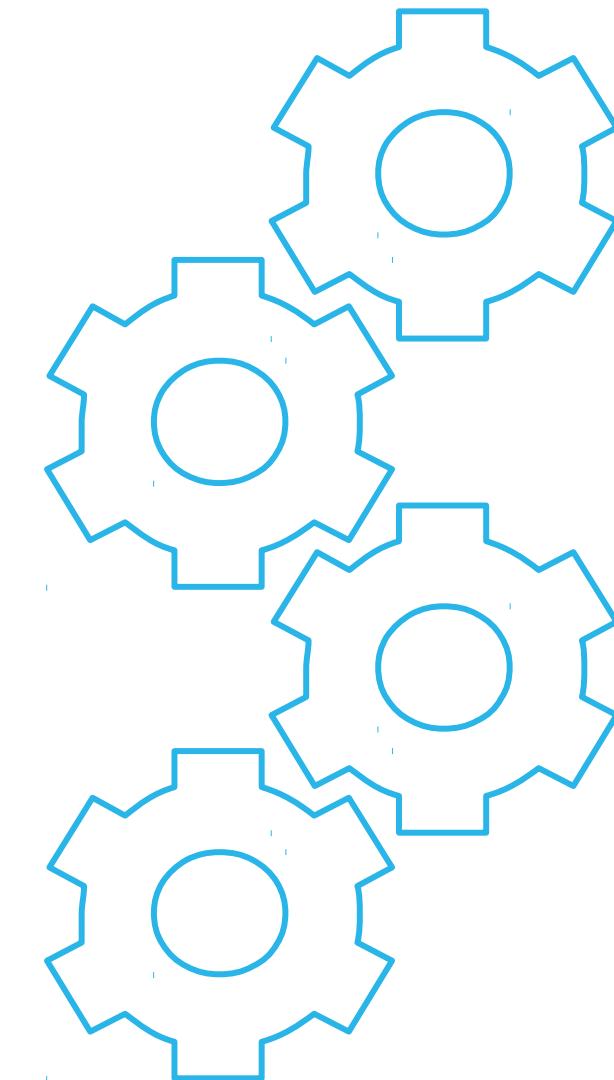
Large



SCALE OUT FOR CONCURRENCY

General Functionality and Considerations

- Single virtual warehouse with multiple compute clusters
- Delivers consistent SLA, automatically adding and removing compute clusters based on concurrent usage
- Scale out during peak times and scale back during slow times
- Queries load balanced across the clusters in a virtual warehouse
- Deployed across availability zones for high availability



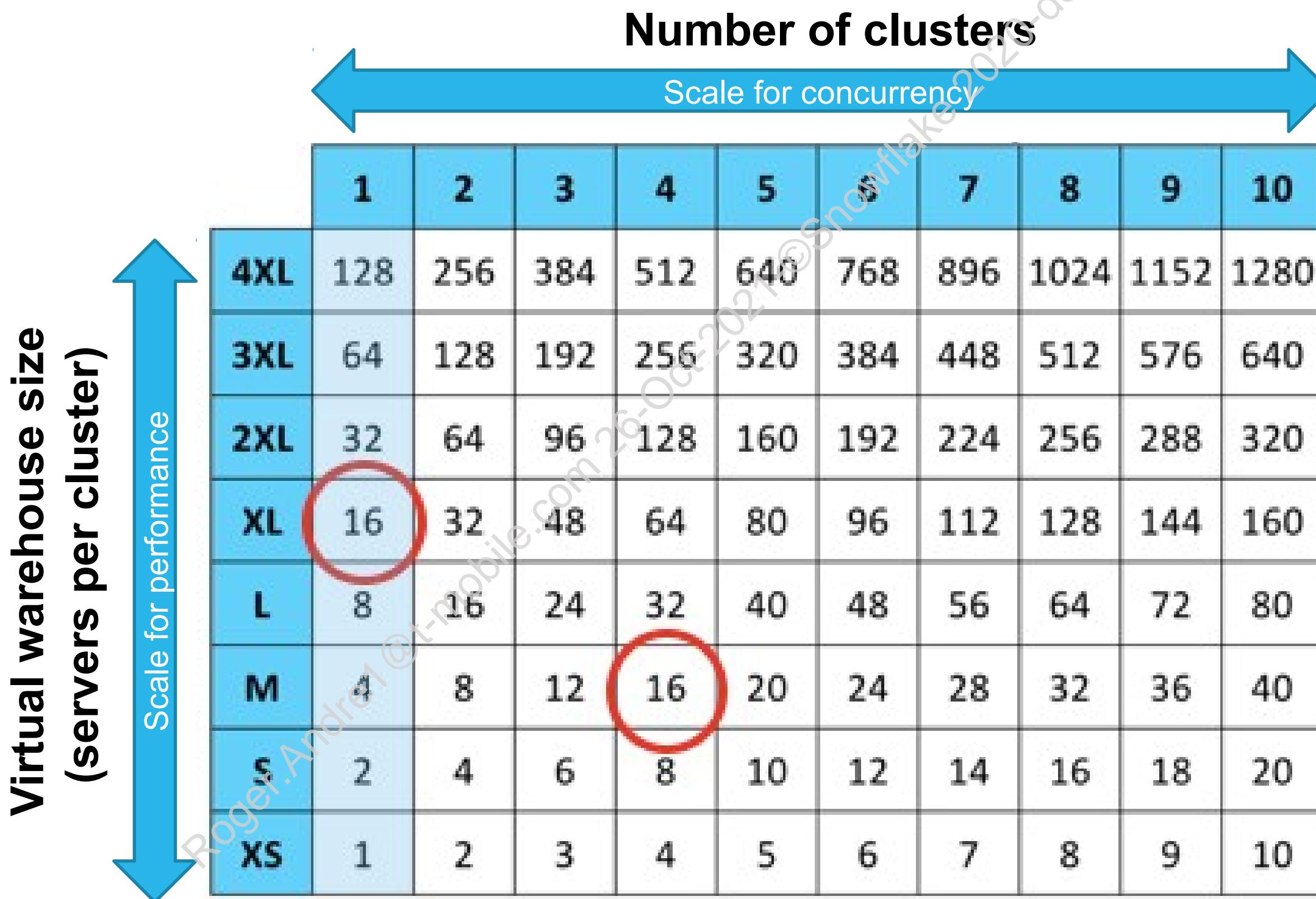
Guidelines

- MIN_CLUSTER_COUNT: 1-10 (default 1)
- MAX_CLUSTER_COUNT: 1-10 (default 1) >= MIN_CLUSTER_COUNT



SCALING UP VS. OUT EXAMPLE

CREDITS PER HOUR



LAB EXERCISE: 2

Work with Storage and Compute

25 minutes

Tasks:

- Review the TRAINING_DB database
- Create and organize objects
- Review storage usage
- Run commands without compute
- Work with virtual warehouses



CLIENTS, CONNECTORS, AND ECOSYSTEMS

Roger.Andre1@t-mobile.com
26-Oct-2021
© 2021 Snowflake Computing Inc. All Rights Reserved
Snowflake 2020-do-not-copy



MODULE AGENDA

- Clients & Interfaces
- SnowSQL
- Connectors
- Ecosystem Overview

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



CLIENTS & INTERFACES

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



WAYS TO CONNECT & USE SNOWFLAKE

- Web-based user interface - access all aspects of using and managing Snowflake
 - For example: WebUI and Snowsight
- Command-line clients - access all aspects of using and managing Snowflake
 - For example: SnowSQL
- Third-party solutions - leverage native connectors to connect to Snowflake
 - For example: ETL and BI tools
- ODBC and JDBC drivers - used by other applications to connect to Snowflake
 - For example: Excel for ODBC and DBeaver for JDBC
- Native connectors - used to develop applications for connecting to Snowflake
 - For example, Python



SNOWFLAKE WEB INTERFACE URLs

AWS US West	Account name	<account>.snowflakecomputing.com
All other regions on AWS		<account>.<region>.snowflakecomputing.com
Other providers	<account>.<region>.<provider>.snowflakecomputing.com	Hostname



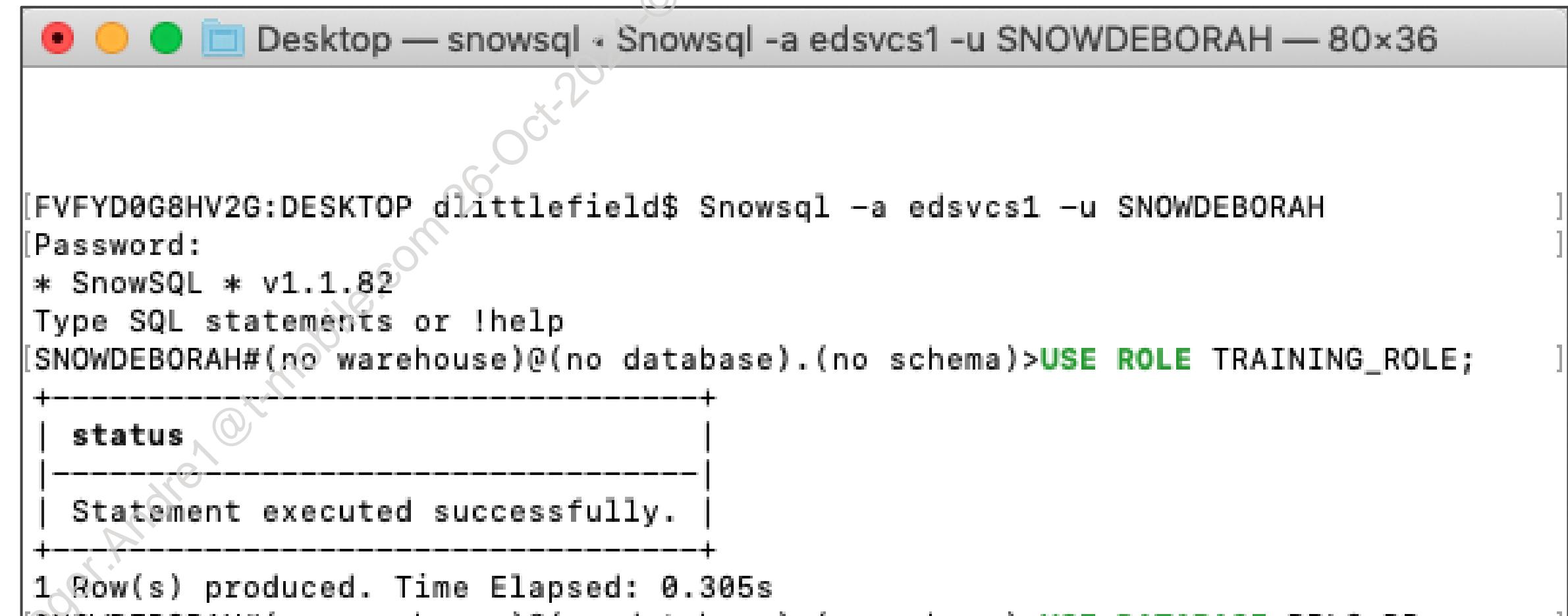
SNOWSQL

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SNOWSQL

- Command-line client for connecting to Snowflake
- Developed using the Snowflake connector for Python
- Versions for Windows, MacOS, Linux



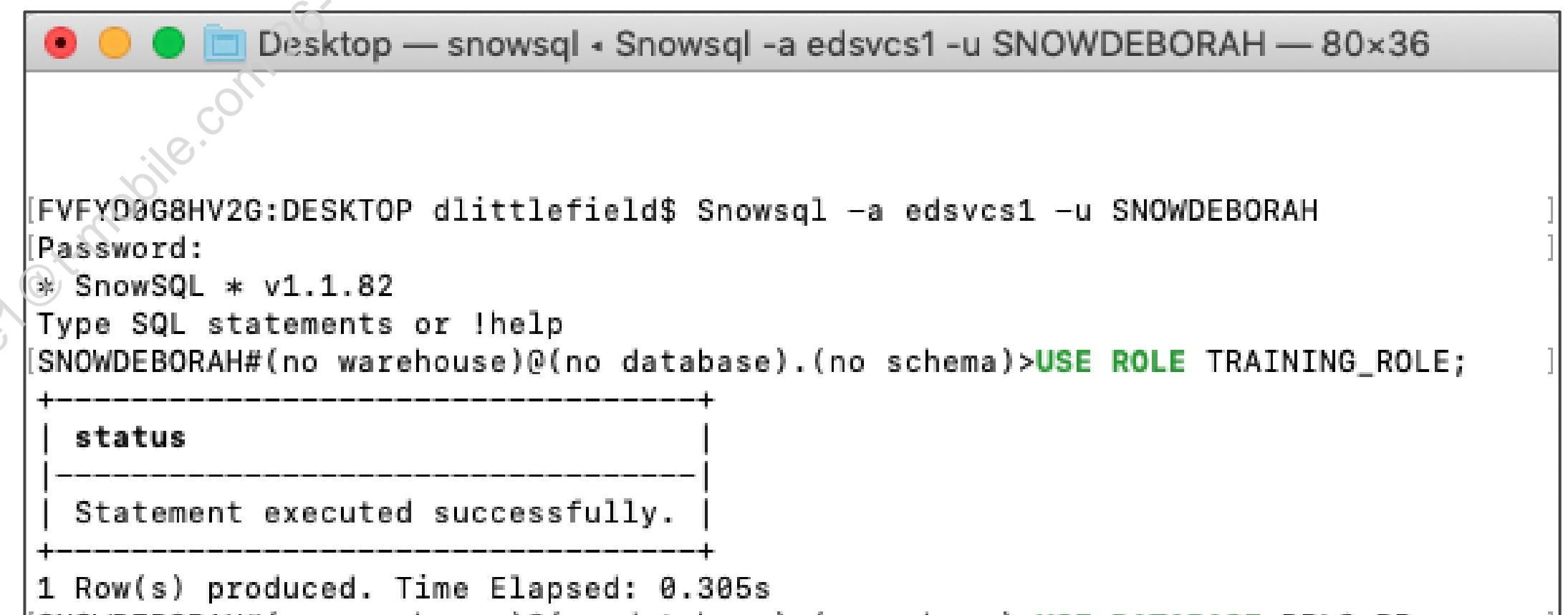
The screenshot shows a terminal window titled "Desktop — snowsql · Snowsql -a edsvcs1 -u SNOWDEBORAH — 80x36". The window displays the following command-line session:

```
[FVFYD0G8HV2G:DESKTOP dlittlefield$ Snowsql -a edsvcs1 -u SNOWDEBORAH
>Password:
* SnowSQL * v1.1.82
Type SQL statements or !help
[SNOWDEBORAH#(no warehouse)@(no database).(no schema)>USE ROLE TRAINING_ROLE;
+-----+
| status
+-----+
| Statement executed successfully.
+-----+
1 Row(s) produced. Time Elapsed: 0.305s]
```



SNOWSQL

- Available for download from the UI: Help --> Downloads
- Run as an interactive shell, or in batch mode
- Can leverage a configuration file to pass parameters like account, user, role, database, schema, etc.
- Can load/unload data to/from local file systems.



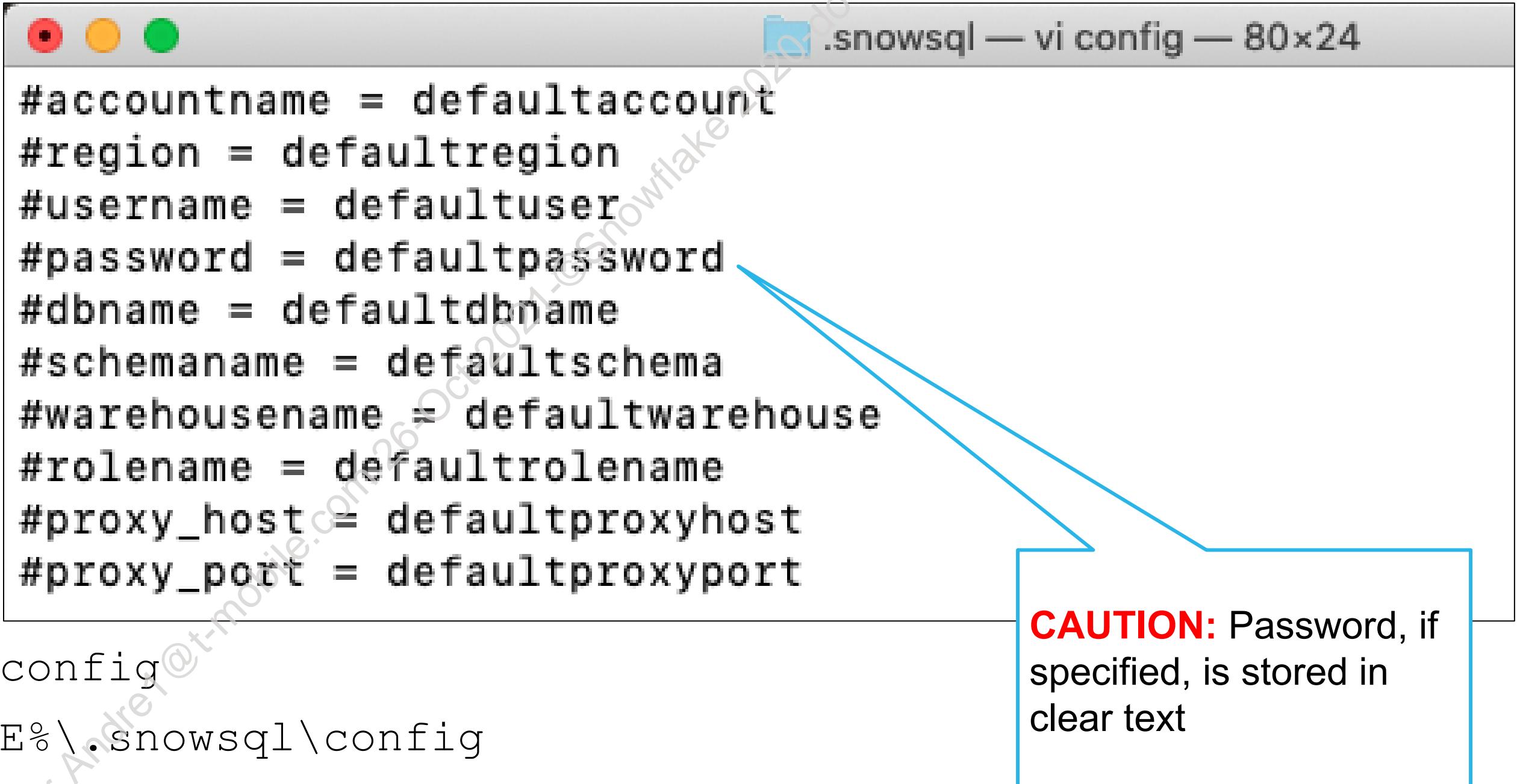
The screenshot shows a terminal window titled "Desktop — snowsql". The command entered was "Snowsql -a edsvcs1 -u SNOWDEBORAH". The output shows the password prompt, the version information ("* SnowSQL * v1.1.82"), the SQL prompt ("[SNOWDEBORAH#(no warehouse)@(no database).(no schema)]>"), and the execution of the "USE ROLE TRAINING_ROLE;" command. The output indicates that the statement was executed successfully and that 1 Row(s) produced. The time elapsed was 0.305s.

```
[FVFYD0G8HV2G:DESKTOP dlittlefield$ Snowsql -a edsvcs1 -u SNOWDEBORAH
>Password:
* SnowSQL * v1.1.82
Type SQL statements or !help
[SNOWDEBORAH#(no warehouse)@(no database).(no schema)]>USE ROLE TRAINING_ROLE;
+-----+
| status
+-----+
| Statement executed successfully.
+-----+
1 Row(s) produced. Time Elapsed: 0.305s
```



SNOWSQL CONFIGURATION FILE

- Set defaults to be used in SnowSQL
- Can set up multiple connectors
- Locations:
 - Linux/Mac: ~/.snowsql/config
 - Windows: %USERPROFILE%\.snowsql\config



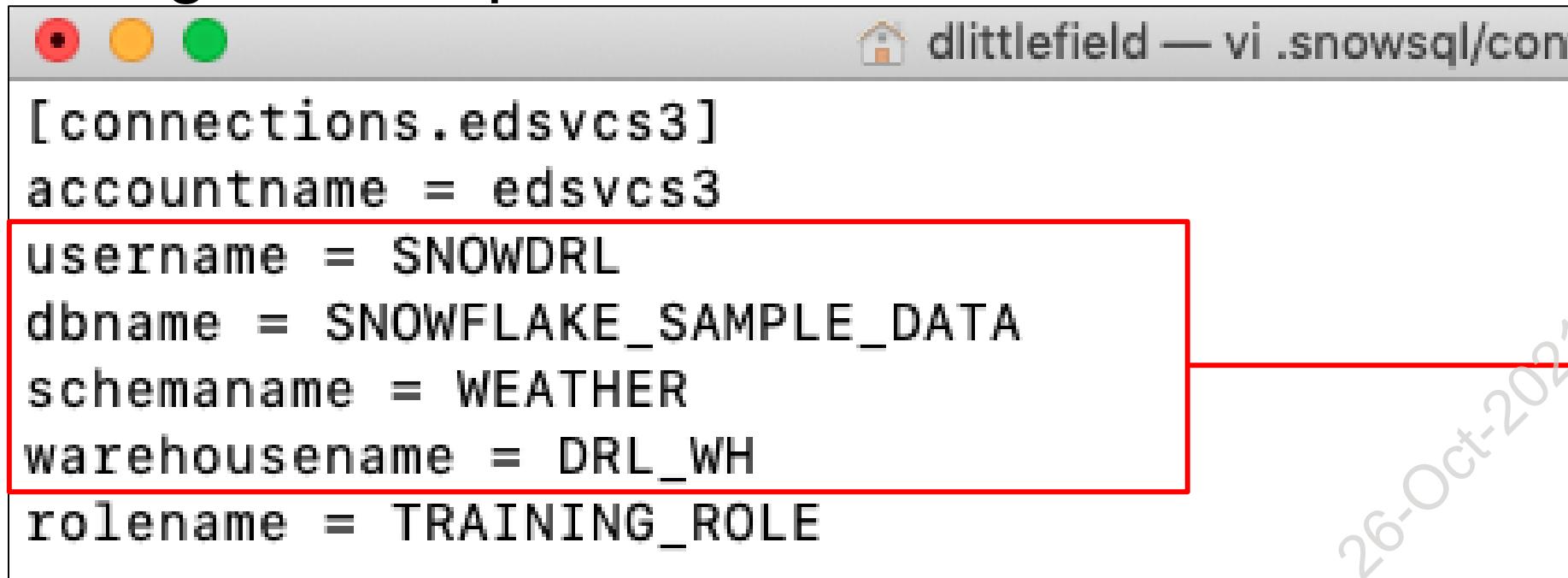
```
#accountname = defaultaccount
#region = defaultregion
#username = defaultuser
#password = defaultpassword
#dbname = defaultdbname
#schemaname = defaultschema
#warehousename = defaultwarehouse
#rolename = defaultrolename
#proxy_host = defaultproxyhost
#proxy_port = defaultproxyport
```

CAUTION: Password, if specified, is stored in clear text



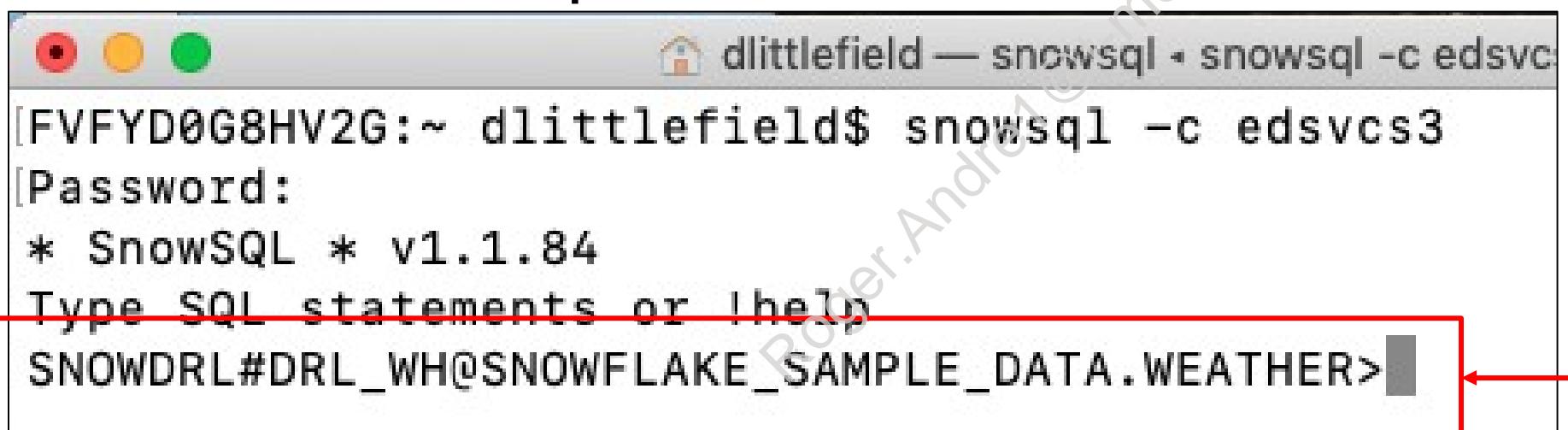
SNOWSQL CONFIGURATION FILE

- Configure multiple connections



```
[connections.edsvcs3]
accountname = edsvcs3
username = SNOWDRL
dbname = SNOWFLAKE_SAMPLE_DATA
schemaname = WEATHER
warehousename = DRL_WH
rolename = TRAINING_ROLE
```

- Connect with -c option



```
[FVFYD0G8HV2G:~ dlittlefield$ snowsql -c edsvcs3
[Password:
* SnowSQL * v1.1.84
Type SQL statements or !help
SNOWDRL#DRL_WH@SNOWFLAKE_SAMPLE_DATA.WEATHER>
```



CHECK CURRENT ROLE IN SNOWSQL

```
[FVFYD0G8HV2G:~ dlittlefield$ snowsql -c edsvcs3
>Password:
* SnowSQL * v1.1.84
Type SQL statements or !help
[SNOWDRL#(no warehouse)@SNOWFLAKE_SAMPLE_DATA.WEATHER>SELECT CURRENT_ROLE();
+-----+
| CURRENT_ROLE() |
|-----|
| TRAINING_ROLE  |
+-----+
1 Row(s) produced. Time Elapsed: 0.125s
SNOWDRL#(no warehouse)@SNOWFLAKE_SAMPLE_DATA.WEATHER>
```



INSTRUCTOR DEMO

SnowSQL

15 minutes



SNOWSIGHT (PUBLIC PREVIEW)

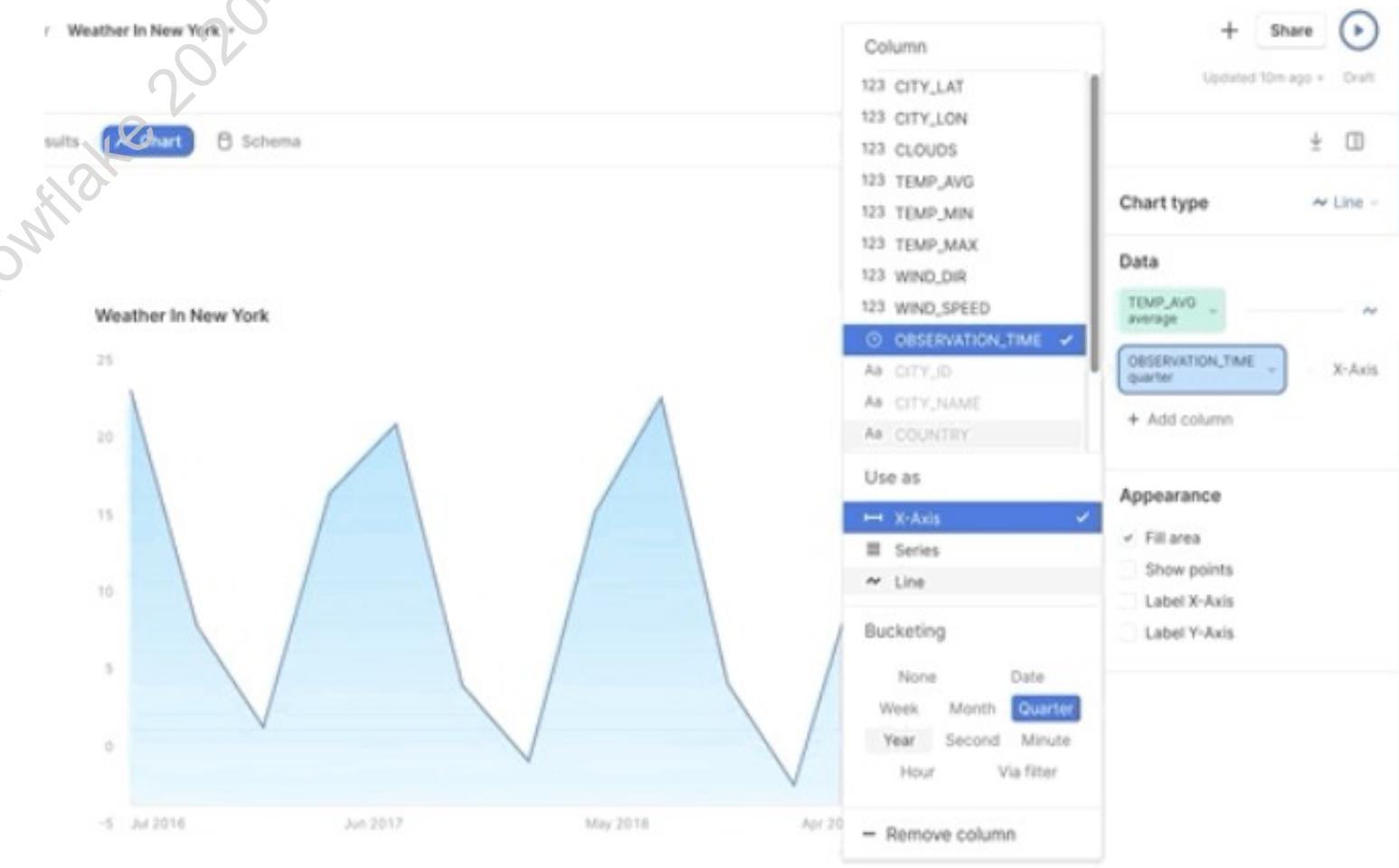
Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SNOWSIGHT

ANALYST EXPERIENCE

- Fast and responsive querying with autocomplete
- Interactive results exploration
- Beautiful visualizations
- Modern dashboards
- Sharing and collaboration
- **NOT** a replacement for your BI tool

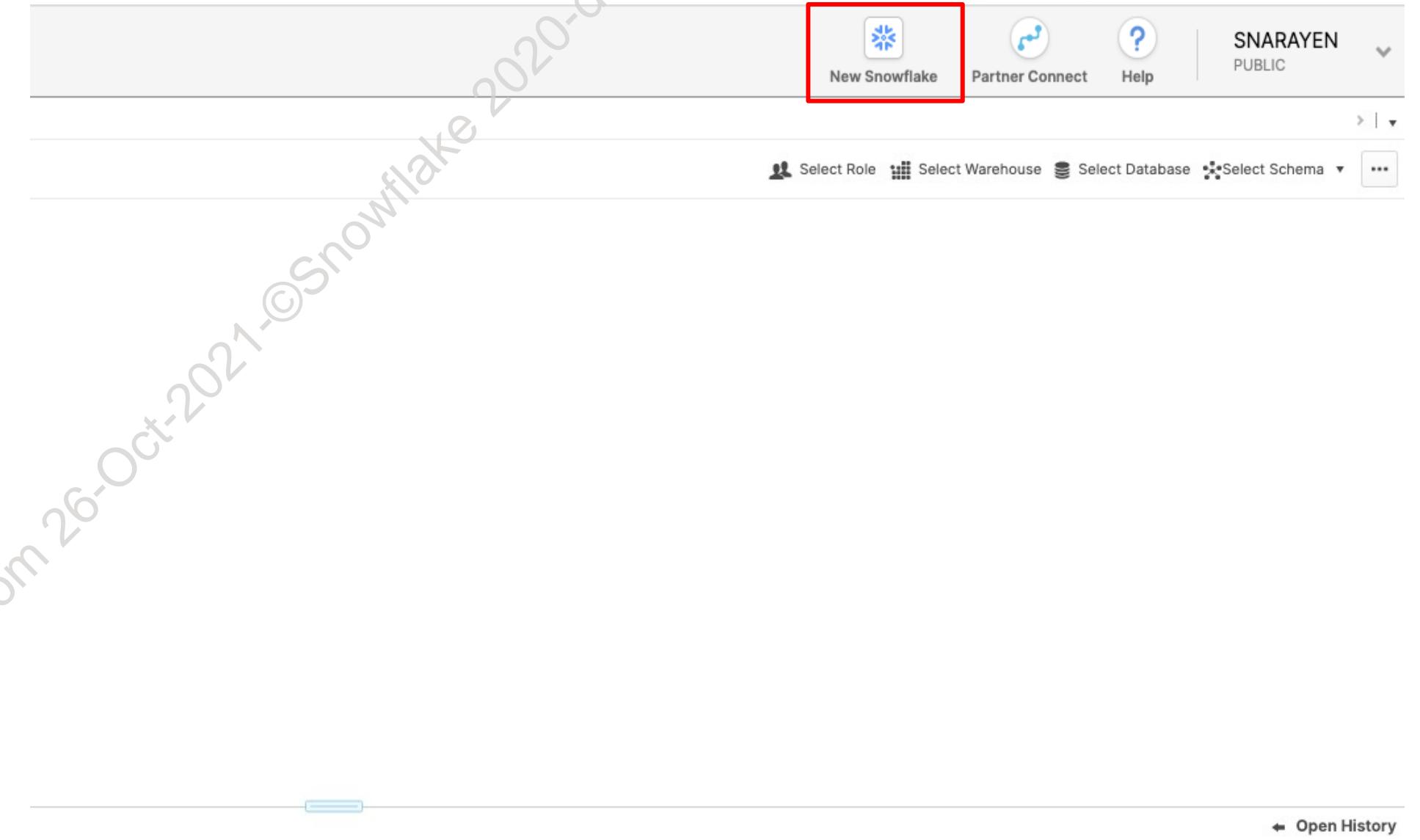


```
8
9
10
11
12
13
14
15
...
select date_trunc('day', starttime) as date,
       count(*) as num_trips,
       t|
     from t
    where
      group|
```



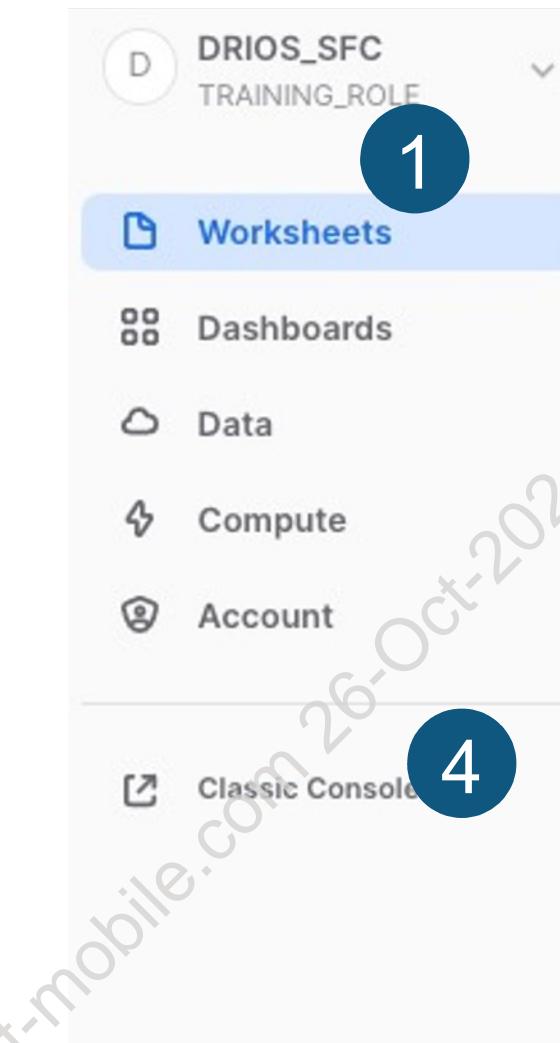
NAVIGATING TO SNOWSIGHT

- A button will appear in the top right of the old UI
- Clicking that button will open the preview UI in a new tab - the preview and old UIs can both be used at the same time
- Access can be granted to individual users or to the entire account



WORKSHEET DIRECTORY - INCLUDING FOLDERS

1 Navigation pane



2 Browse worksheets and folders

A screenshot of the Snowflake Worksheets page. The title 'Worksheets' is at the top, followed by tabs for Recent, Shared with me, My Worksheets, and Folders. A search bar and a 'Worksheet' button are on the right. The main area shows a table of worksheets with columns for TITLE, VIEWED (sorted descending), and UPDATED. The table contains the following data:

TITLE	VIEWED ↓	UPDATED
AVG WINDOWING TEST	4 days ago	4 days ago
SUM GROSS REV	5 days ago	5 days ago
Semi-structured lab demo	6 days ago	6 days ago
Capstone lab demo	1 week ago	1 week ago
Time travel lab demo	1 week ago	1 week ago
Caching and Query Perf lab demo	1 week ago	1 week ago
TPCH Queries	1 week ago	1 month ago

3 Add a new worksheet, or import worksheets from Classic Console

4

4 Return to Classic Console



WORKSHEETS OVERVIEW

- 1 Choose the role, warehouse, and database for this worksheet
- 2 Query editor with intelligent text formatting
- 3 Switch between browsing objects, writing queries, viewing results and creating charts
- 4 Run your query (or use command-enter)
- 5 View the worksheet's query history

The screenshot shows the Snowflake Worksheet interface with the following numbered elements:

- 1 Top right corner showing role (TRAINING_ROLE), warehouse (CAMEL_WH), and a share button.
- 2 Query editor displaying a SELECT statement. The code is:

```
SNOWBEARAIR_DB.PROMO_CATALOG_SALES_STAR ▾  
1 SELECT  
2   D.ORDER_DATE  
3   , SUM(GROSS_REVENUE) AS SUM_GROSS_REVENUE  
4 FROM  
5   ORDER_DATE_DIM D  
6   INNER JOIN REVENUE_FACT RF ON D.DATE_ID = RF.ORDER_DATE_ID  
7 WHERE  
8   RF.RETURN_FLAG = 'N'  
9 GROUP BY  
10  D.ORDER_DATE;
```
- 3 Results table showing the output of the query. The columns are ORDER_DATE and SUM_GROSS_REVENUE. The data is:

	ORDER_DATE	SUM_GROSS_REVENUE
1	2012-09-30	481,107.51
2	2013-08-27	491,586.81
3	2013-10-07	476,670.79
4	2015-02-18	524,999.6
5	2017-01-21	985,280.35
6	2016-09-26	988,477.47
7	2012-04-13	495,522.96
8	2015-05-21	811,195.36
9	2013-09-18	475,209.14
10	2013-09-19	459,421.48
11	2012-11-02	499,787.4

- 4 Top right corner showing the query was updated 4 minutes ago.
- 5 History panel showing the run times for the last two runs: 2:53 PM and 9:31 AM.



EDITING WORKSHEETS

- 1 Autocomplete shows available columns after defining an alias for a database

The screenshot shows a Snowflake worksheet interface. The query being typed is:

```
SNOWBEARAIR_DB.PROMO_CATALOG_SALES_STAR ▾  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13
```

The cursor is at position 7, after the comma in the first column of the SELECT statement. A dropdown menu is open, listing available columns from the ORDER_DATE_DIM table:

- DATE_ID
- column number
- DISCOUNT
- column number
- DISCOUNT_PER_PART
- column number
- D
- alias Alias of ORDER_DATE_DIM

The 'Objects' tab is selected in the bottom navigation bar.

- 2 Use function autocomplete (with a link to Snowflake docs) to quickly write queries

The screenshot shows a Snowflake worksheet interface. The query being typed is:

```
SNOWBEARAIR_DB.PROMO_CATALOG_SALES_STAR ▾  
SELECT  
    D.ORDER_DATE  
    , SUM
```

The cursor is at position 10, after the comma in the second column of the SELECT statement. A dropdown menu is open, listing the SUM function:

- sum([DISTINCT] <expr1>)
- function Returns the sum of non-NULL records for expr Docs

The 'Query' tab is selected in the bottom navigation bar.



BROWSING TABLES

- 1 Pin tables for quick reference to see column names and data types

The screenshot shows the Snowflake interface with the following details:

- The database is set to PROMO_CATALOG_SALES.
- The schema is set to PUBLIC.
- The table being viewed is CUSTOMER.
- A context menu is open over the CUSTOMER table, listing options: Show Details, Place Name in SQL, and Pin.
- The 'Pin' option is highlighted with a blue background.
- Other visible tables in the list include LINEITEM, LINEITEM_AMERICA, LINEITEM_APAC, and LINEITEM_EMEA.

- 2 Preview data from each table

The screenshot shows a preview of the CUSTOMER table data. The table has 150K rows and contains the following columns:

	C_CUSTKEY	C_FIRSTNAME	C_LASTNAME	C_NATIONKEY	C_ACCTBAL
1	108,415	Wolfgang	Owens	11	4,695
2	44,115	Cecilia	Goto	9	1,691
3	139,674	Cory	Thornton	22	3,380
4	15,387	Kim	Choi	14	5,265
5	30,266	Nicholas	Estuardo	19	2,525
6	9,366	Albert	Bloom	8	7,065
7	60,479	Ben	Jalal	3	9,596
8	22,803	Larry	Dixon	12	4,514
9	73,944	Ned	Labong	5	4,054
10	8,189	Yuka	Bakalov	0	8,964
11	19,858	Ulysses	Damon	21	8,167
12	28,422	Thomas	Elliott	18	1,983
13	107,426	Kess	Ortiz	21	1,737
14	29,690	Christian	Espinosa	24	7,354
15	68,527	Quinn	Kimura	4	7,585
16	46,436	Becca	Hagiwara	4	7,076
17	124,745	Tamara	Reed	10	1,320

Below the preview, the table definition is shown:

```
CUSTOMER
150K Rows
C_CUSTKEY
C_FIRSTNAME
C_LASTNAME
C_NATIONKEY
C_ACCTBAL
```

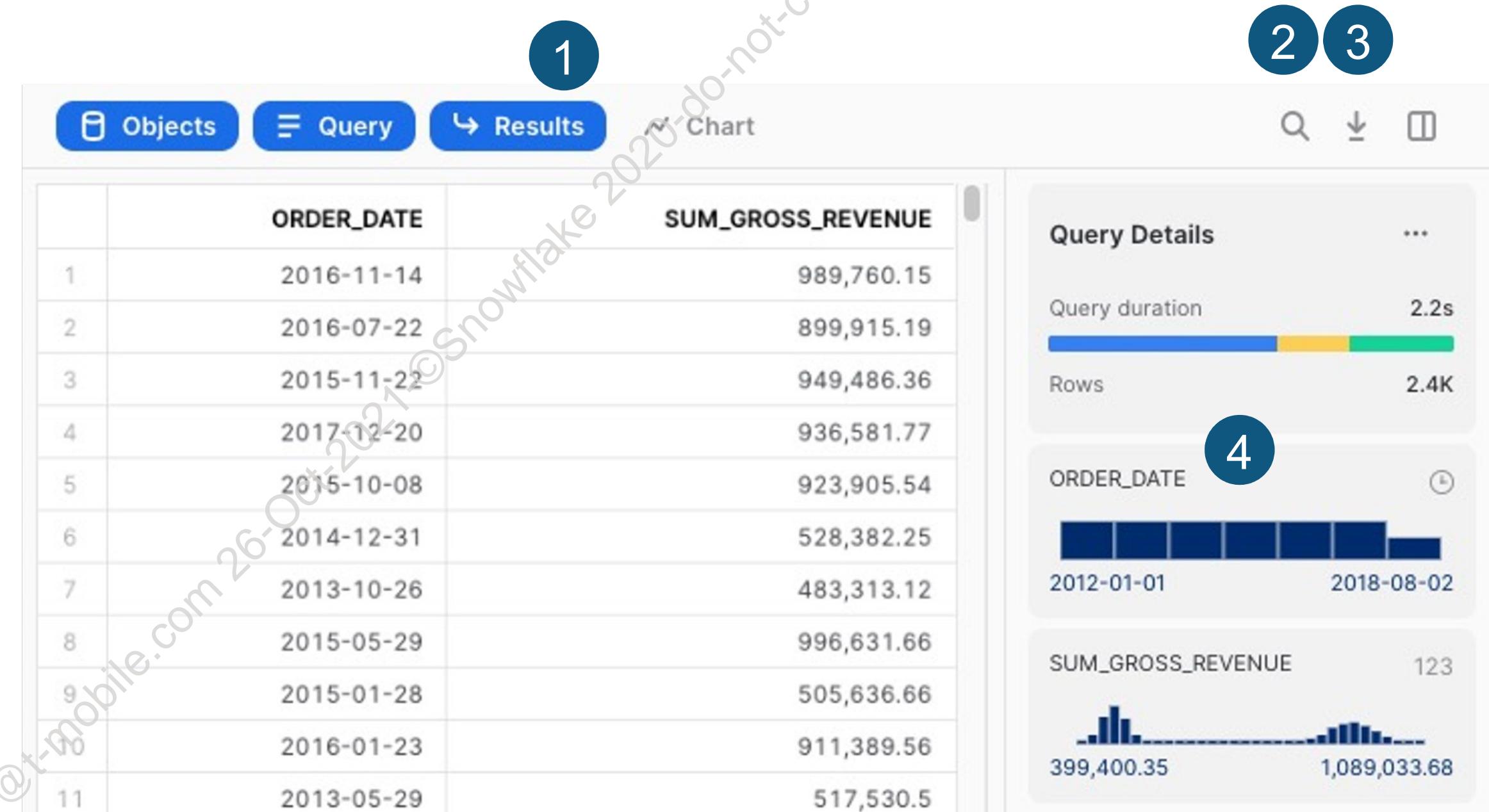
At the bottom right, there is a smaller preview of the C_NATIONKEY column values for Europe:

11	3	EUROPE
12	3	EUROPE
13	3	EUROPE
14	3	EUROPE
15	3	EUROPE
16	3	EUROPE



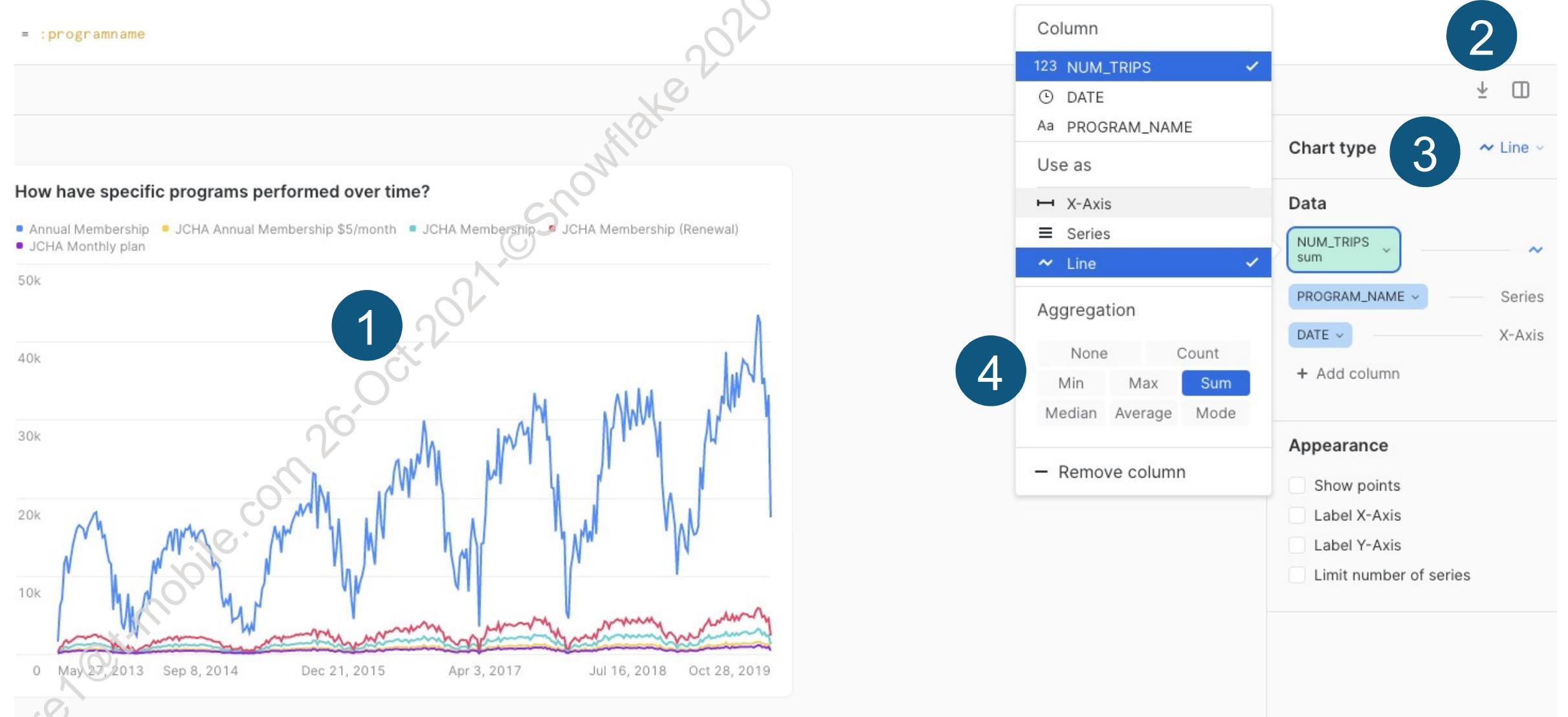
RESULTS

- 1 View your results, sort by any column, and change formatting
- 2 Search across columns and rows to filter your results
- 3 Download your results as a .csv
- 4 Narrow your dataset further by filtering to specific values or ranges in each column



CHARTS

- 1 Visualize and interact with your data
- 2 After you configure the perfect chart, download it as a .png
- 3 Choose between line, bar, scatter, heatmap, or scorecard chart types
- 4 Configure the data source, axes, aggregations, or bucketing with a single click



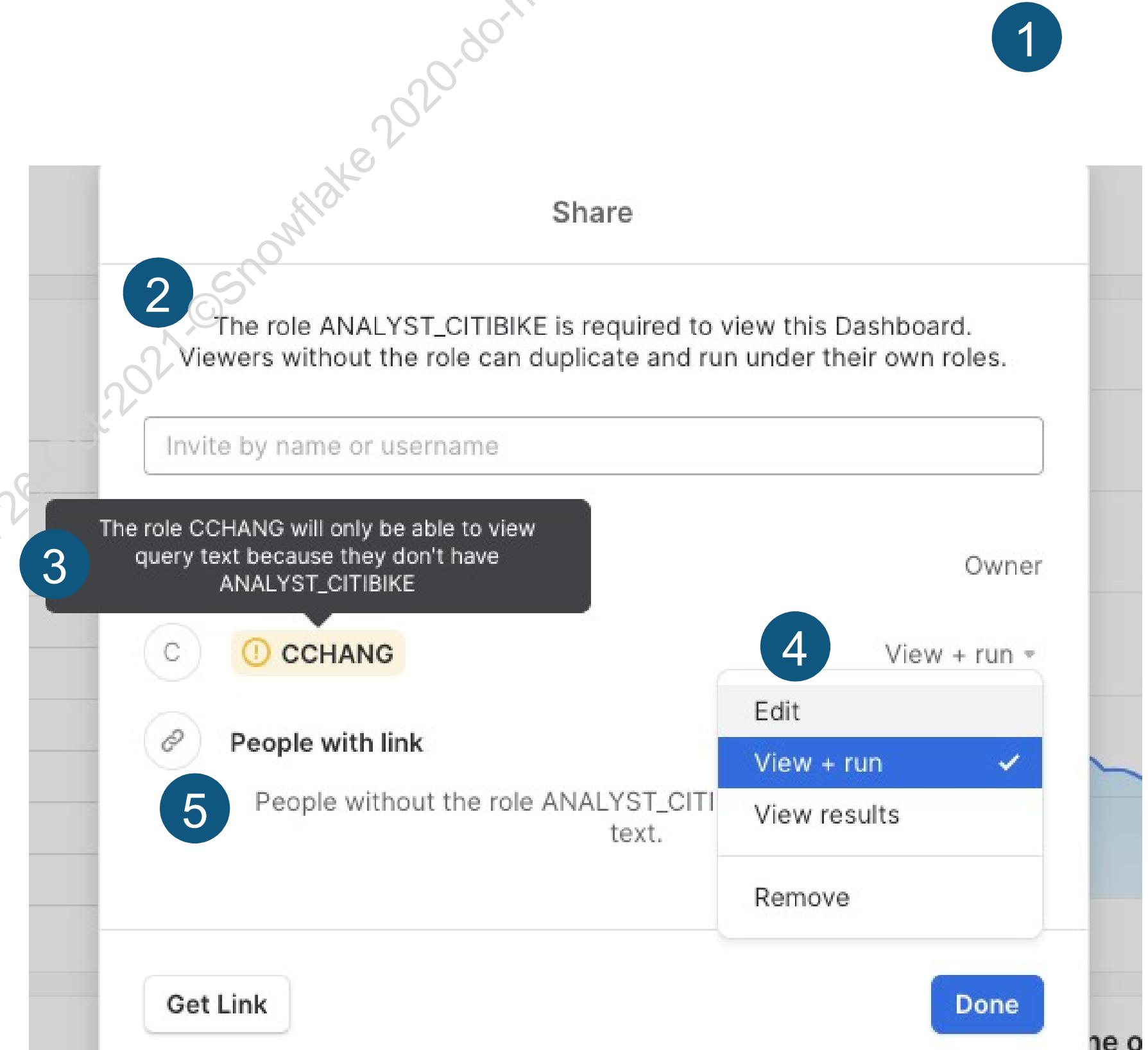
DASHBOARDS

- 1 Add charts to your dashboard from new or existing worksheets
- 2 Choose your role for the dashboard
- 3 Share your dashboard with colleagues



SHARING WORKSHEETS AND DASHBOARDS

- 1 Click the “Share” button in the top right of the screen
- 2 To conform with Snowflake RBAC, we currently limit sharing to other users with the same role
- 3 We proactively warn the user when attempting to share with someone with the wrong role
- 4 Change permissions on that worksheet or dashboard for the shared user
- 5 Change permissions for that worksheet or dashboard for users who receive the link



SNOWFLAKE DOCS

Documentation is available at <https://docs.snowflake.com/en/user-guide/ui-snowsight.html>



Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy

LAB EXERCISE: 3

Explore Snowsight

45 minutes

Exercise: Practice using Snowsight to gain insights into data

Tasks:

- Connecting to Snowsight and working with worksheets
- Working with query history
- Using folders to organize worksheets
- Filtering data
- Creating charts and dashboards
- Downloading charts



CONNECTORS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SUPPORTED CONNECTORS

JDBC	Java
ODBC	C/C++
Python	Python
Go	Go
Node.js	Node.js
Spark	Spark Dataframe (Scala, Python)
R	Java, C, C++ (RJDBC, RODBC)
.NET	Visual Basic, C#, F#, C++
SnowSQL	CLI client for SQL (Python)
Web Interface	UI for Admin, SQL, and more



ECOSYSTEM OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 © snowflake 2020-do-not-copy



 = PARTNER CONNECT

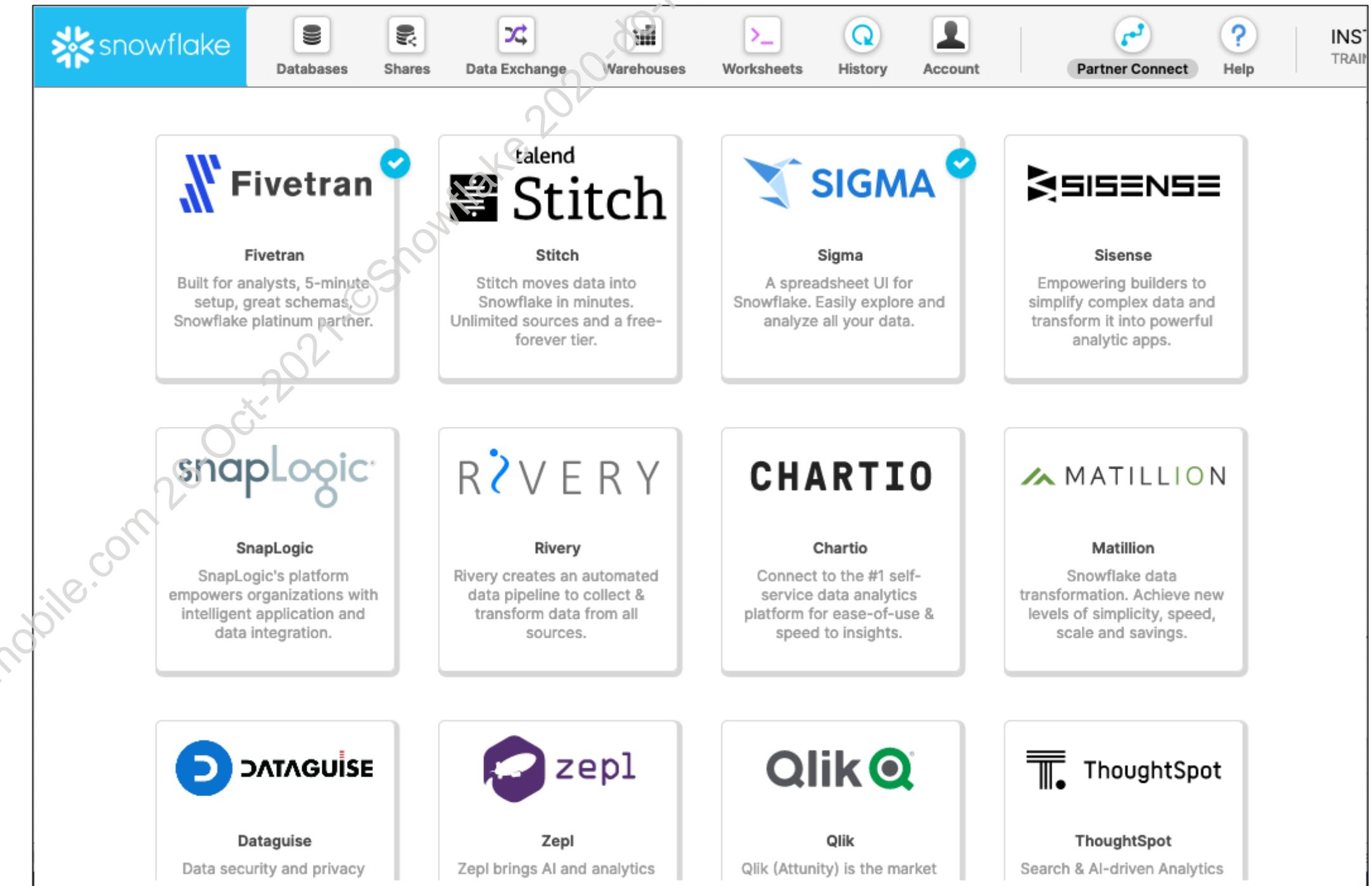


 = PARTNER CONNECT



PARTNER CONNECT

- Create trial account with selected Snowflake partners
- Use limited to ACCOUNTADMIN
- During connection, partner creates:
 - PC_%partnername_USER
 - PC_%partnername_ROLE
 - PC_%partnername_DB
 - PC_%partnername_WH



INSTRUCTOR DEMO

Partner Connect 5 minutes

The screenshot shows the Snowflake Partner Connect interface. At the top, there's a navigation bar with icons for Databases, Shares, Warehouses, Worksheets, History, Account, Preview App, Partner Connect (which is highlighted), Help, and a user login dropdown. The main title is "Snowflake Partner Connect". Below it, a sub-header says "Get started with loading and analyzing your data in minutes. Automatically connect your Snowflake account with our partner applications available for a free trial." A note below that says "Check back often as we will be adding new partners regularly." The page displays ten partner cards arranged in two rows of five:

- Fivetran**: Built for analysts, 5-minute setup, great schemas, Snowflake platinum partner.
- talend Stitch**: Stitch moves data into Snowflake in minutes. Unlimited sources and a free-ever tier.
- SIGMA**: A spreadsheet UI for Snowflake. Easily explore and analyze all your data.
- Sisense**: Empowering builders to simplify complex data and transform it into powerful analytic apps.
- snapLogic**: SnapLogic's platform empowers organizations with intelligent application and data integration.
- Rivery**: Rivery creates an automated data pipeline to collect & transform data from all sources.
- CHARTIO**: Connect to the #1 self-service data analytics platform for ease-of-use & speed to insights.
- MATILLION**: Snowflake data transformation. Achieve new levels of simplicity, speed, scale and savings.
- DATAGUISE**: Data security and privacy automation for compliance with GDPR, CCPA, PCI and more.
- zepl**: Zepl brings AI and analytics to your Snowflake data in minutes. Try it for free today.



SNOWFLAKE CACHING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Overview
- Metadata Cache
- Query Result Cache
- Data Cache

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

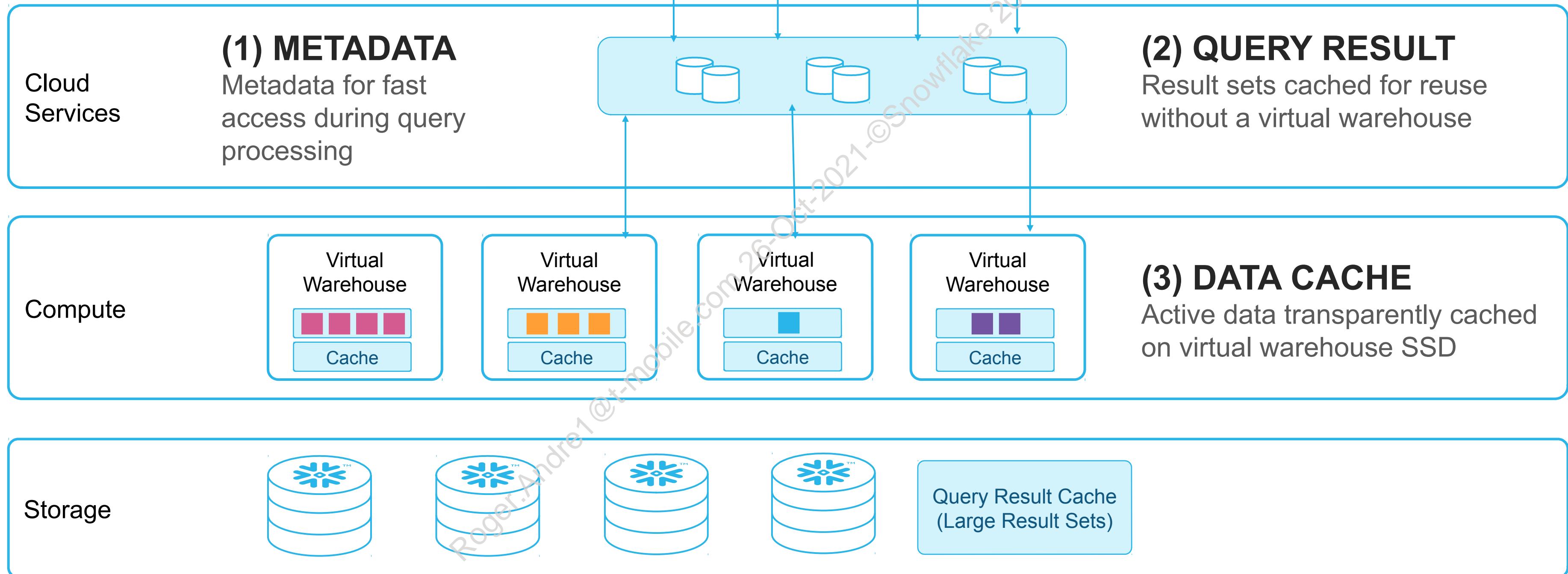


OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



CACHING IN SNOWFLAKE

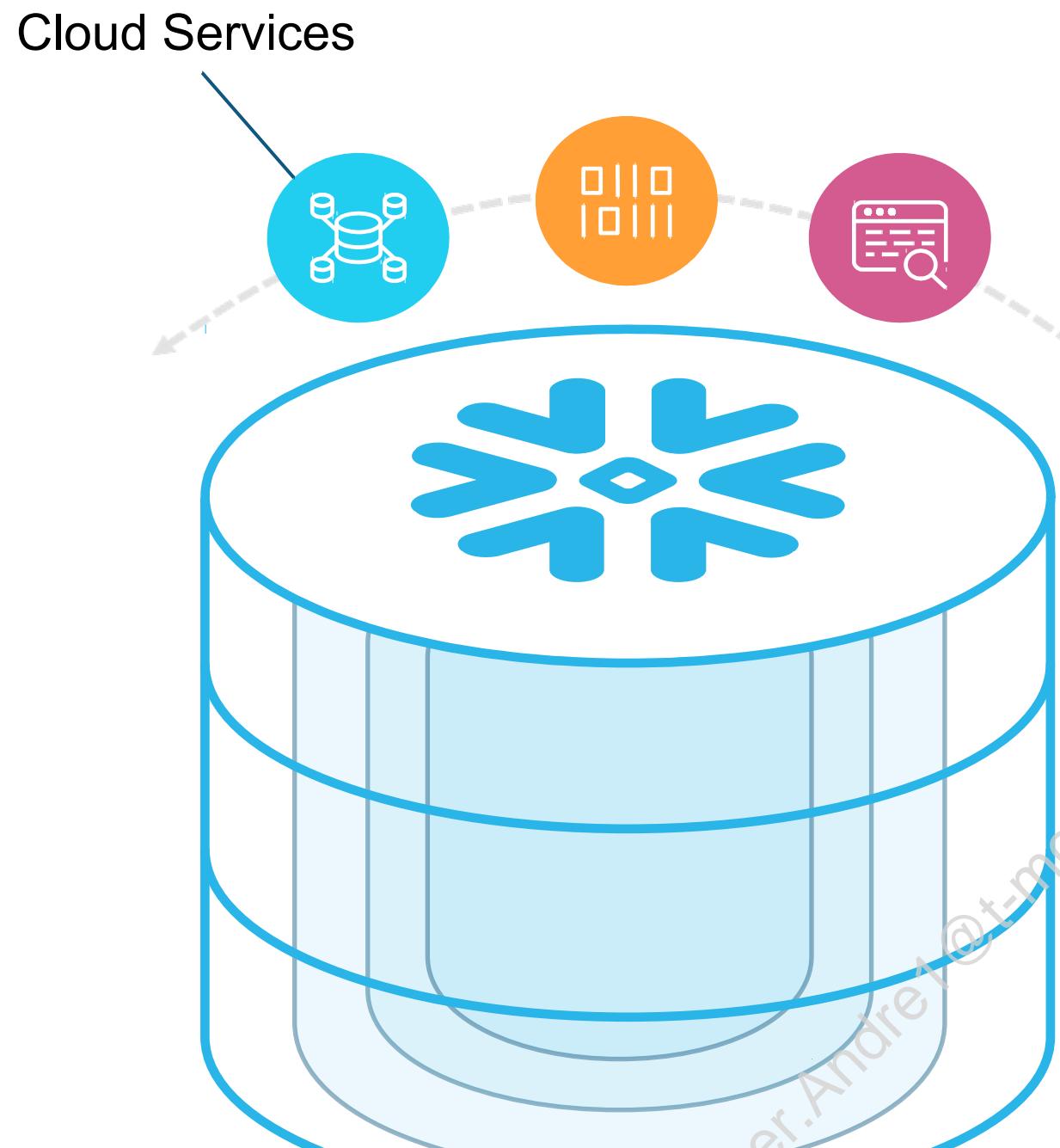


METADATA CACHE

Roger.Andre1@t-mobile.com 26-Oct-2021 @snowflake 2020-do-not-copy



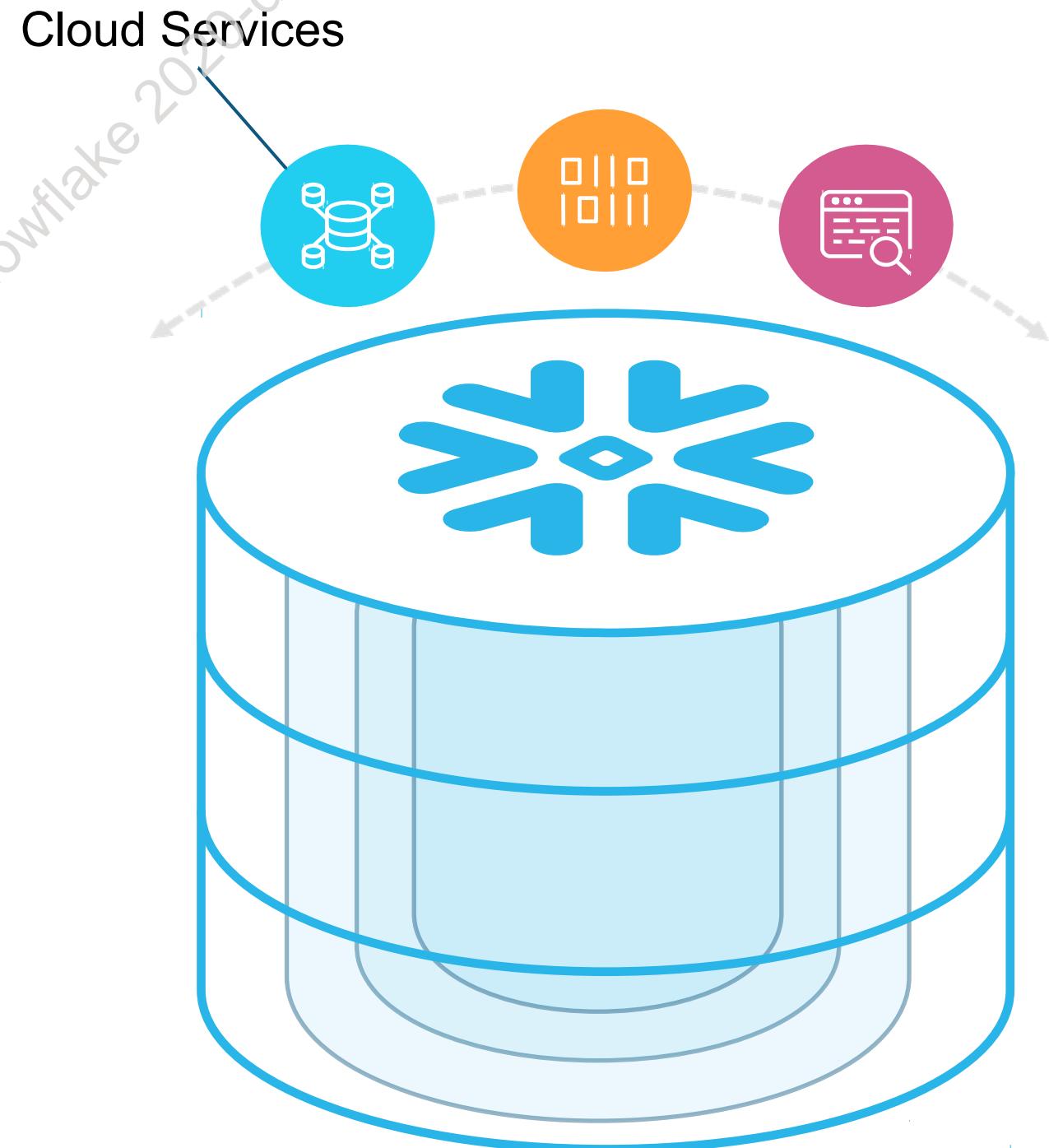
METADATA CACHE



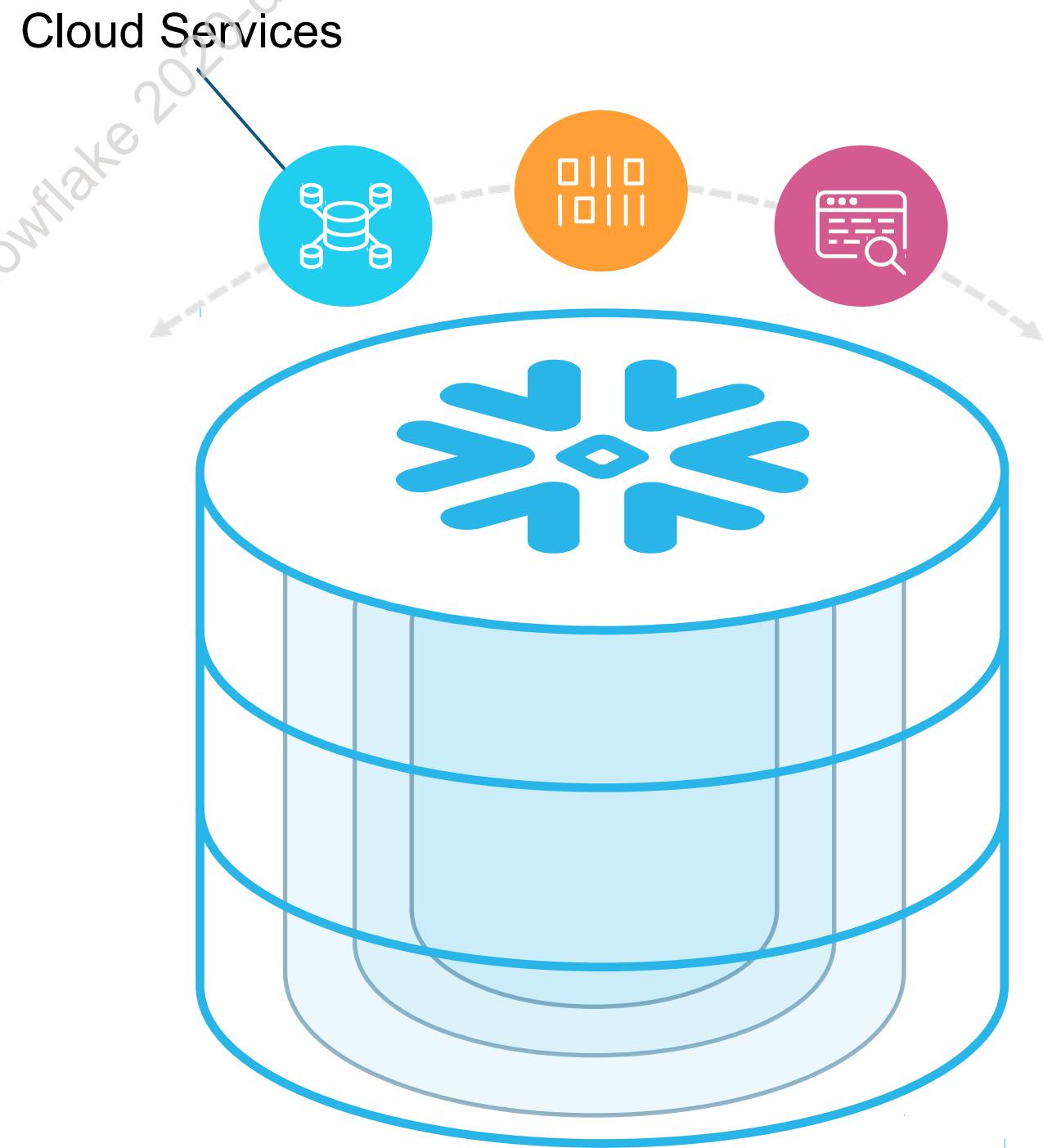
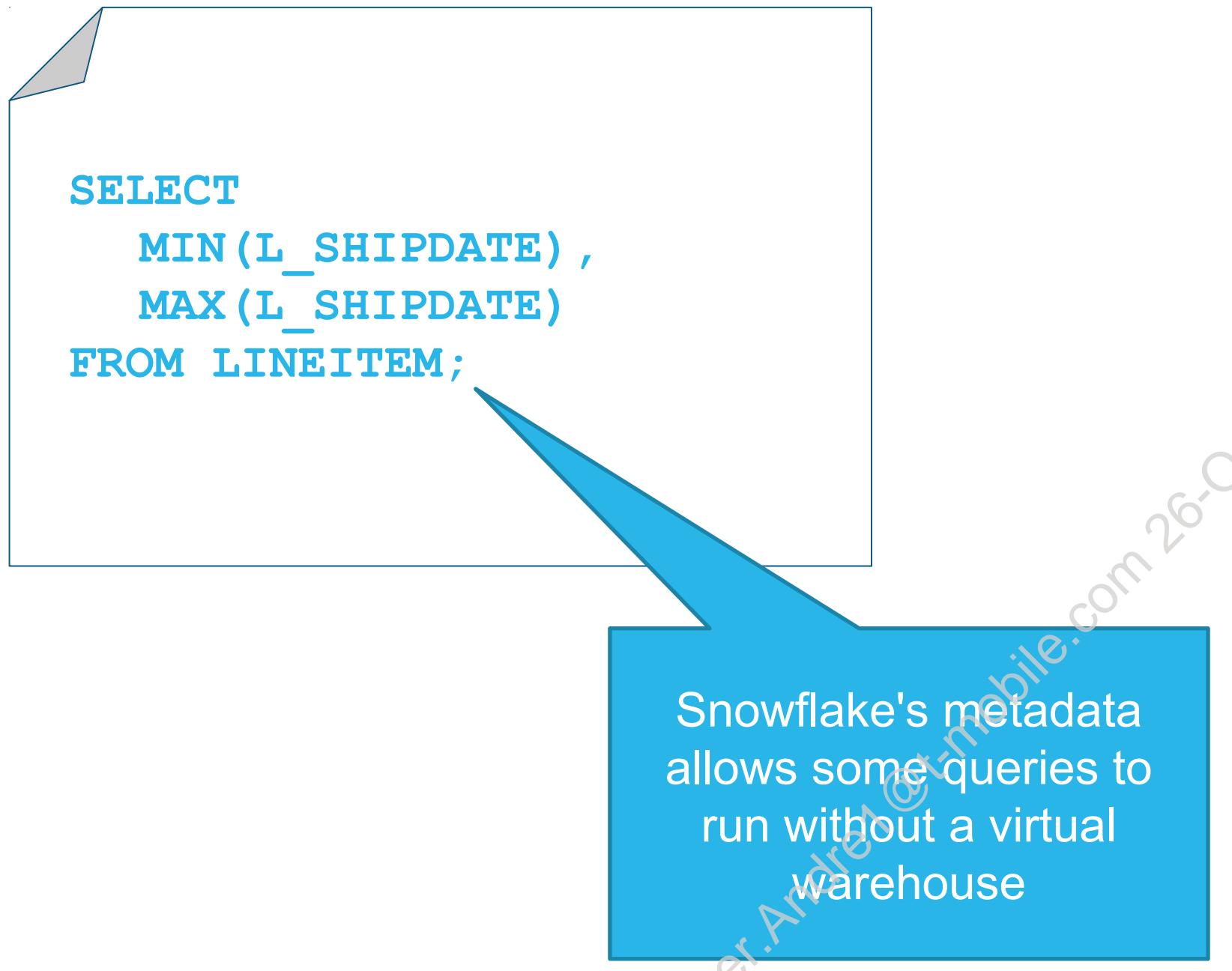
- Metadata stored in cloud services layer
- Micro-partition level metadata
 - Row count
- Micro-partition column-level metadata
 - MIN/MAX values
 - Number of DISTINCT values
 - Number of NULL values
- Table versions and references to physical files (.fdn)

MICRO-PARTITION METADATA CACHE

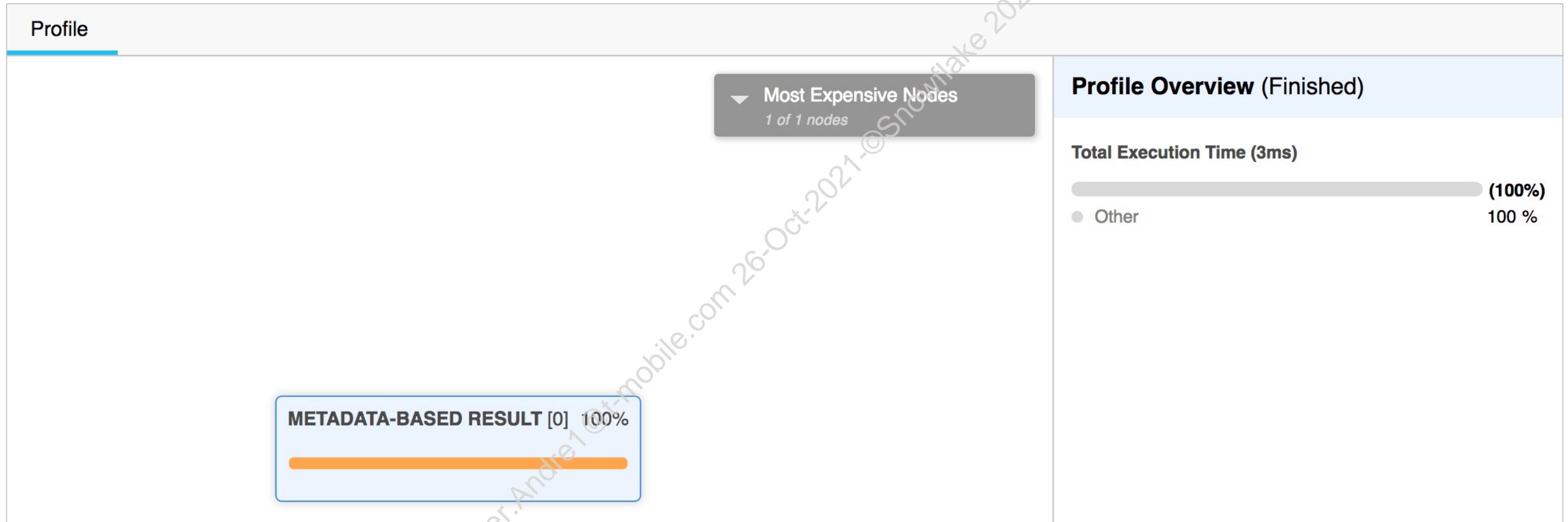
- Used by SQL optimizer to speed up query compilation
- Used for queries that can be answered completely by metadata
 - SHOW commands
 - MIN, MAX (only for integer and date data types)
 - COUNT
- Fast
- No virtual warehouse used
 - NOTE: Cloud Services charges may still apply!



METADATA CACHE – QUERY EXAMPLE



METADATA CACHE - QUERY PROFILE



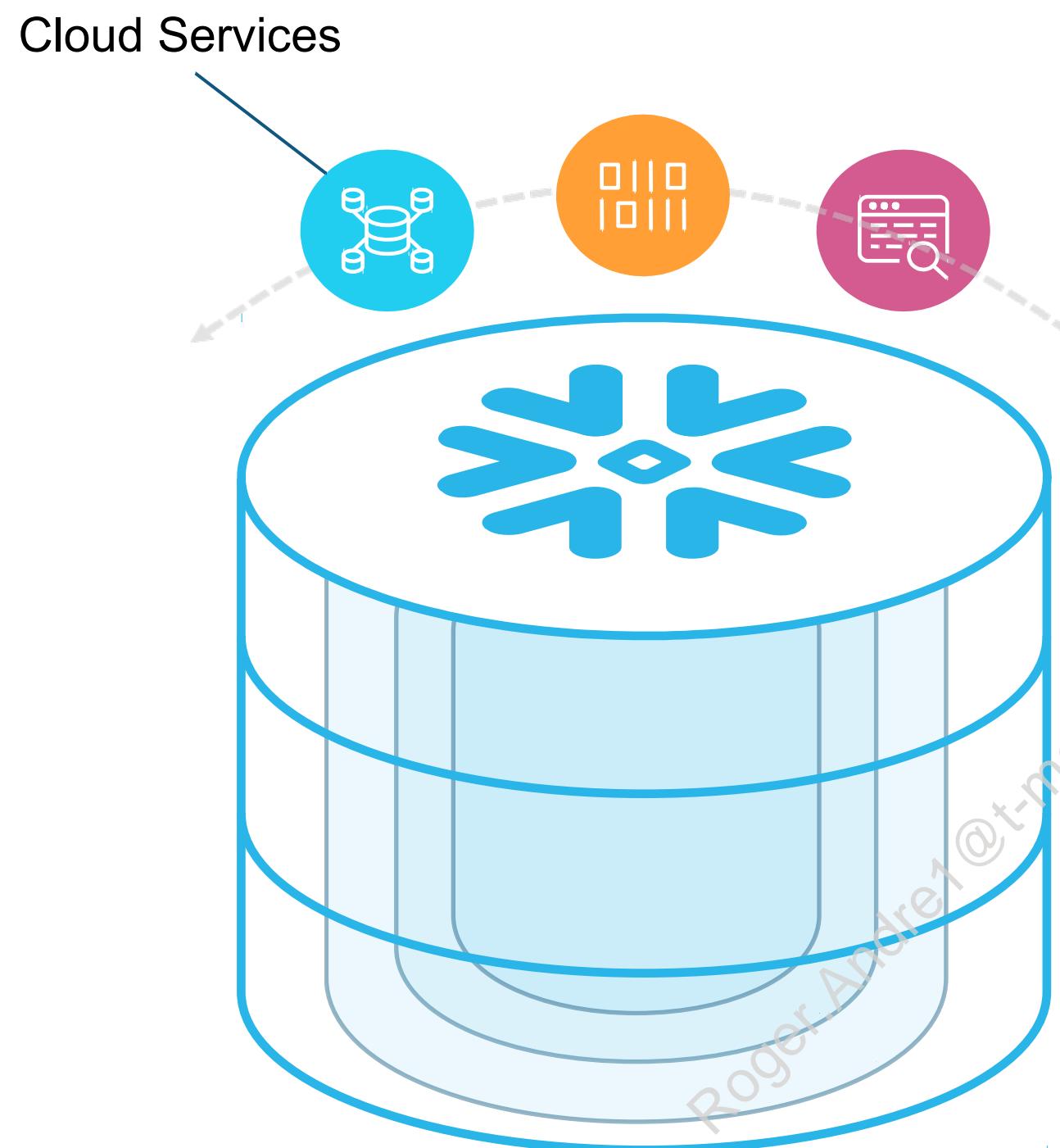
QUERY RESULT CACHE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy



QUERY RESULT CACHE

PERSISTED QUERY RESULTS

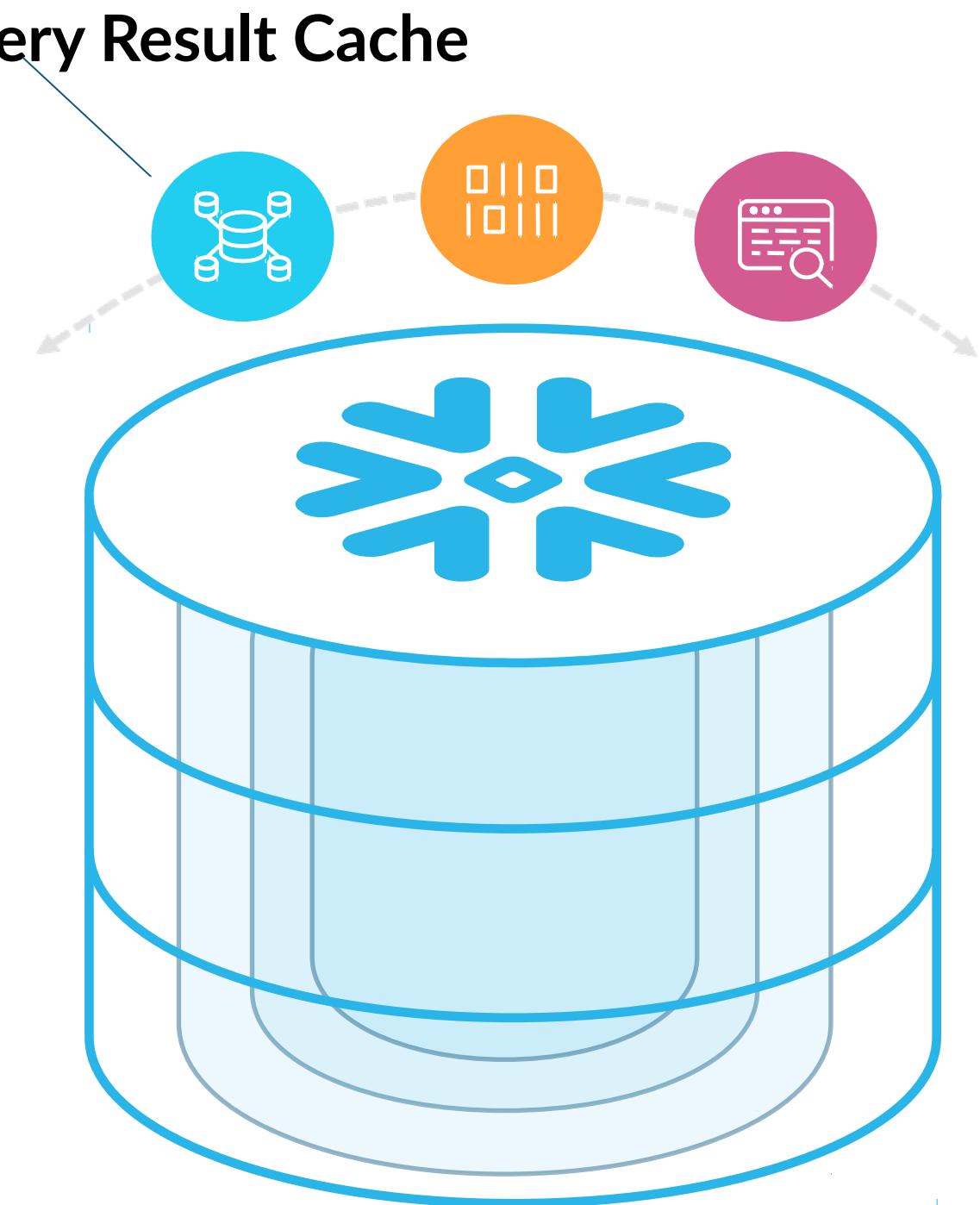


- Query results are stored and managed by the Cloud Services layer
- Used if the identical query is run, and base tables have not changed
- Available to other users:
 - SHOW: any user in the same role
 - SELECT: any user with SELECT permissions on all tables in query



HOW IT WORKS

- Result sets are cached for 24 hours; counter resets each time matching query is re-used
- Result reuse controlled by `USE_CACHED_RESULT` parameter at account/user/session level
- Eligibility requirements for query to use result set cache:
 - Exact same SQL query (* except maybe whitespace)
 - Result must be deterministic (no random function)
 - Changes CAN be made to source table(s), but only if they do not affect any micro-partitions relevant to query
 - Must have the right permissions to use it

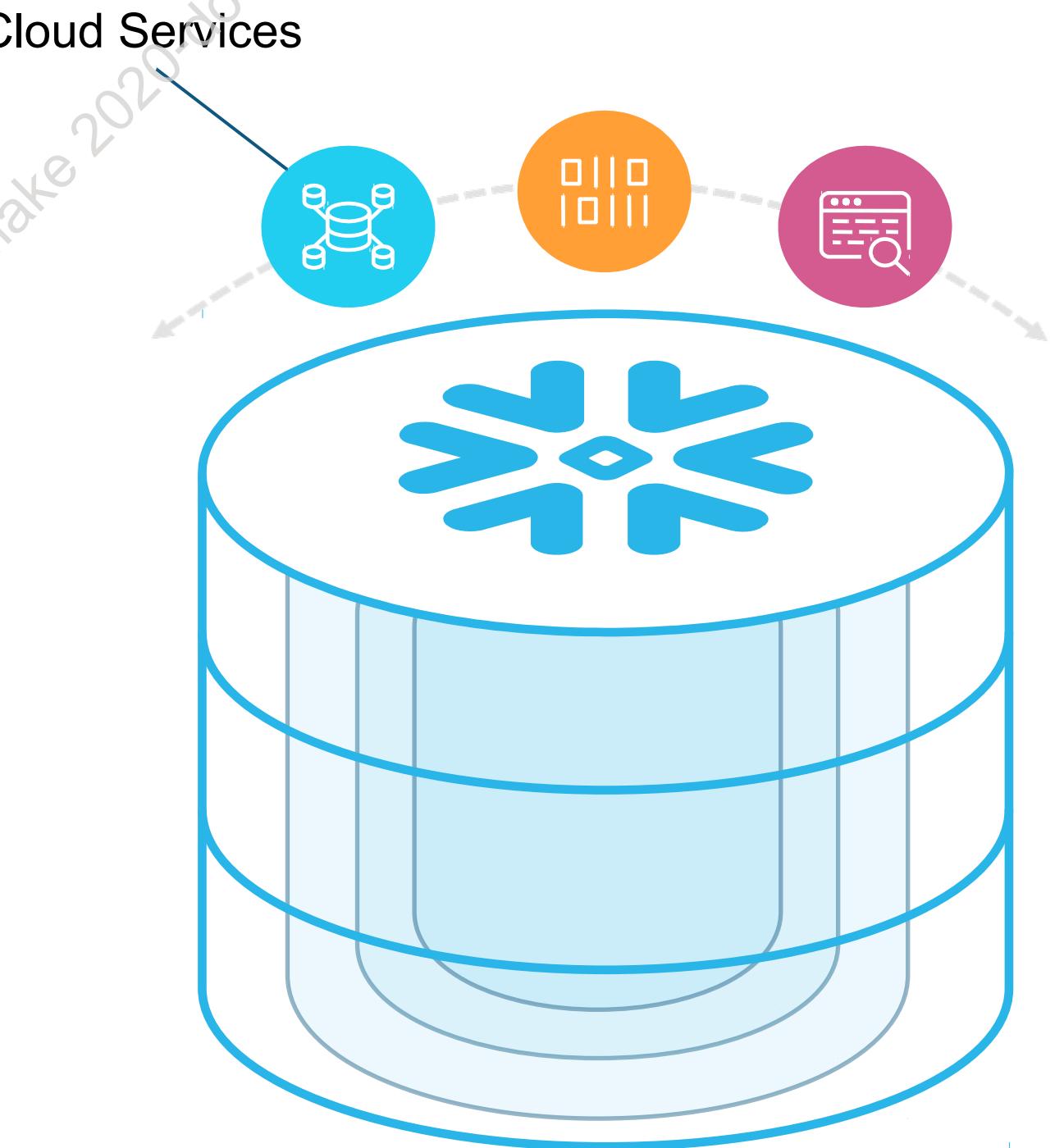


QUERY RESULT CACHE USE CASES

- Static dashboards
- Queries with significant computing
 - Semi-structured data analysis
 - Aggregates
- Queries that are run frequently or are complex
- Refine the output of another query
 - Use TABLE function `RESULT_SCAN (<query_id>) ;`

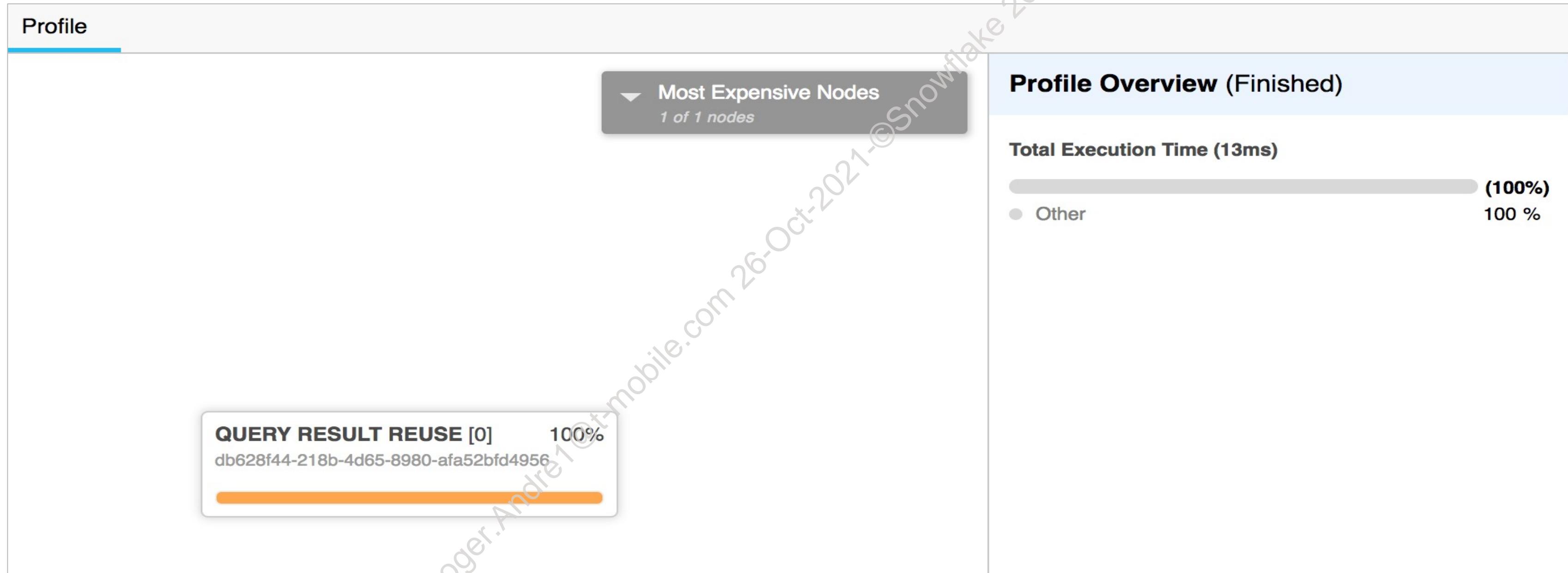
Benefits

- Fast
- Will never give stale results
- No virtual warehouse used



QUERY RESULT CACHE EXAMPLE

Query profile confirms re-use of query result cache

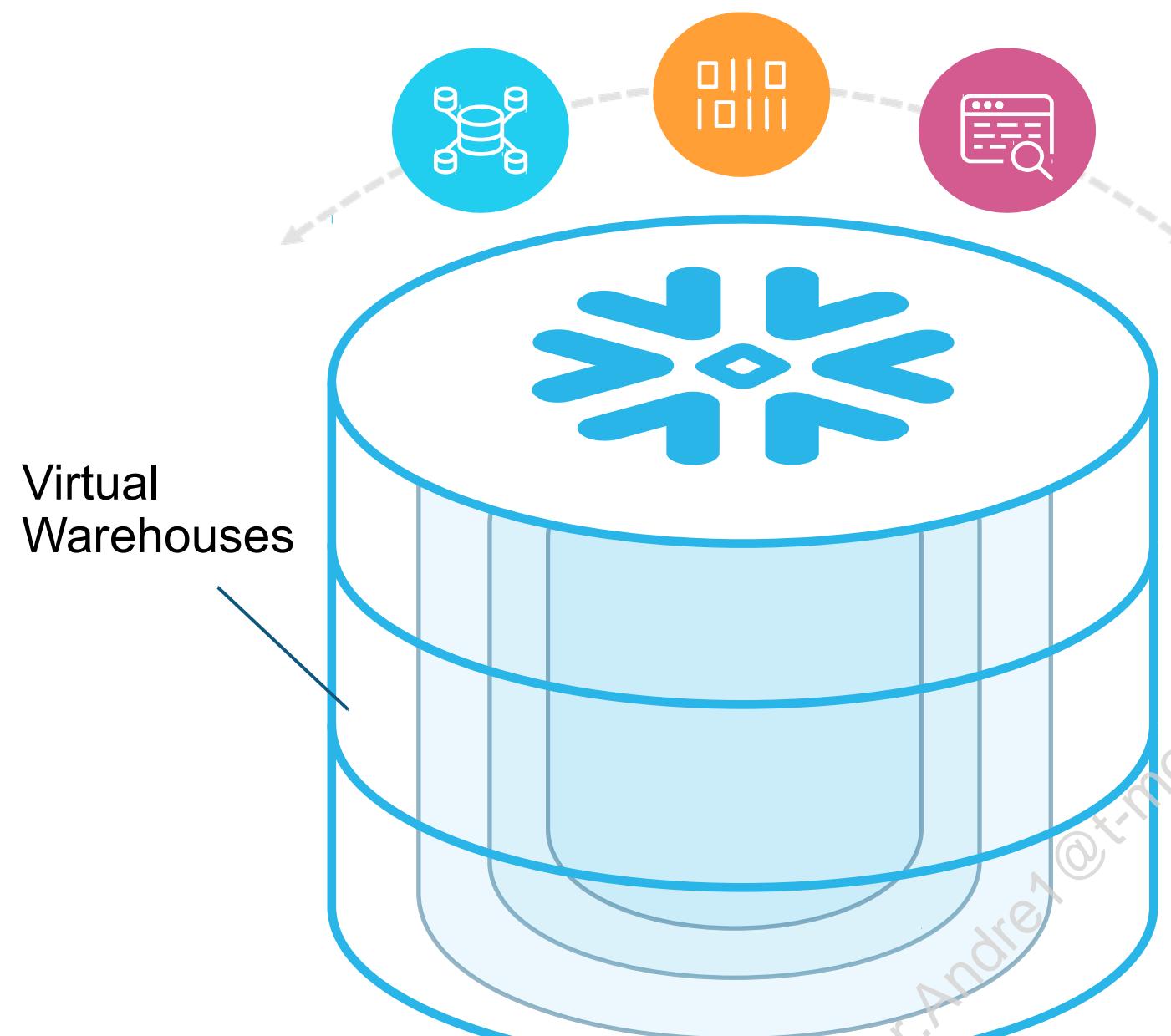


DATA CACHE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA CACHE



- Stores file headers and column data from queries
 - Stores the data, not the result
- Stored to SSD in virtual warehouse
- When a similar query is run, Snowflake will use as much data from the cache as possible
- Available for all queries run on the same virtual warehouse



DATA CACHE

Query Example:

```
SELECT network FROM rating WHERE (data_stream = 'Live');
```

What is in the Data Cache?

- The partitions needed to satisfy the query
 - Columns in the SELECT clause
 - Columns in the WHERE clause

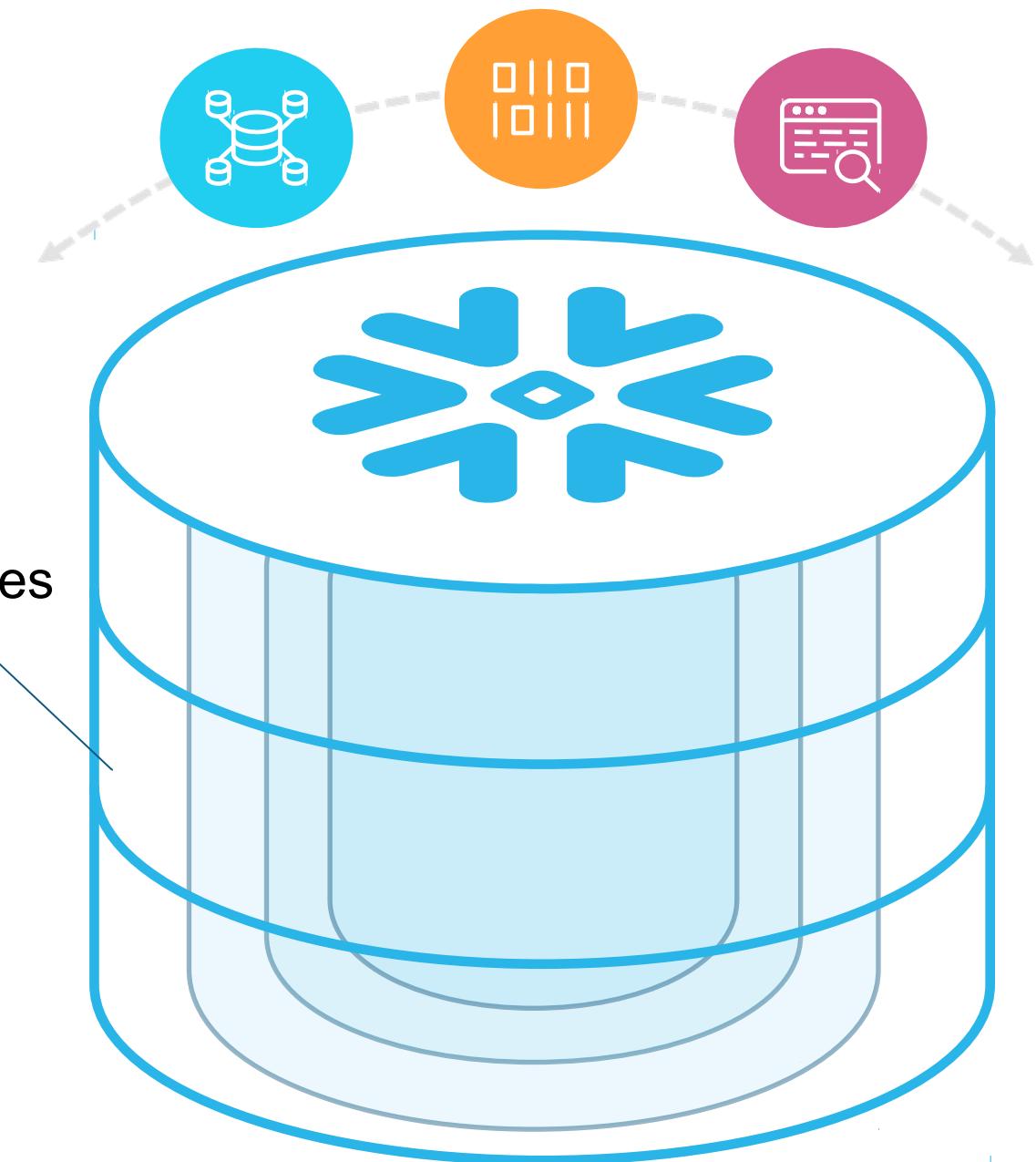
Best Practice:

- Group and execute similar queries on the same virtual warehouse to maximize data cache reuse, for performance and cost optimization

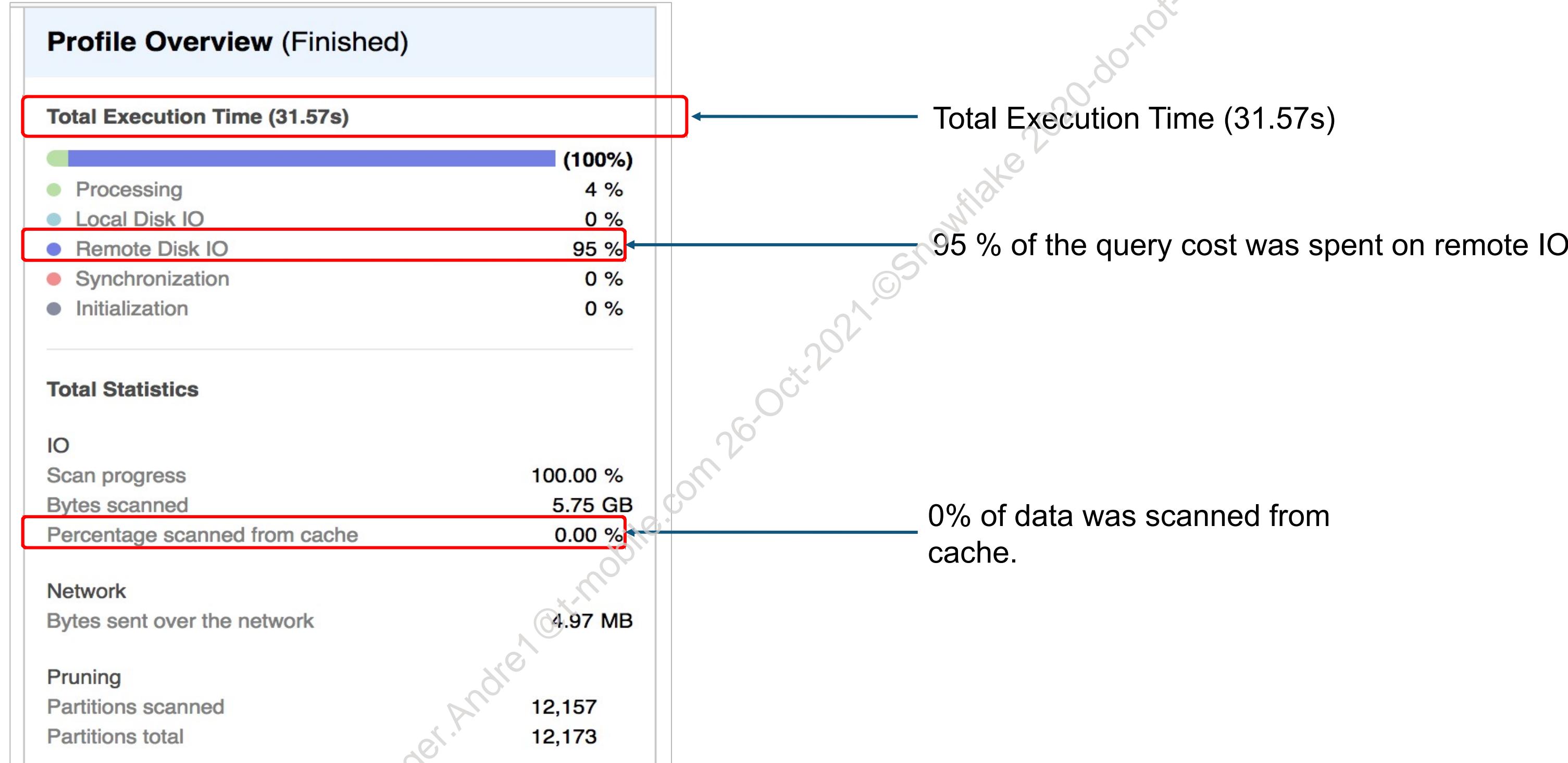


HOW DATA CACHE WORKS

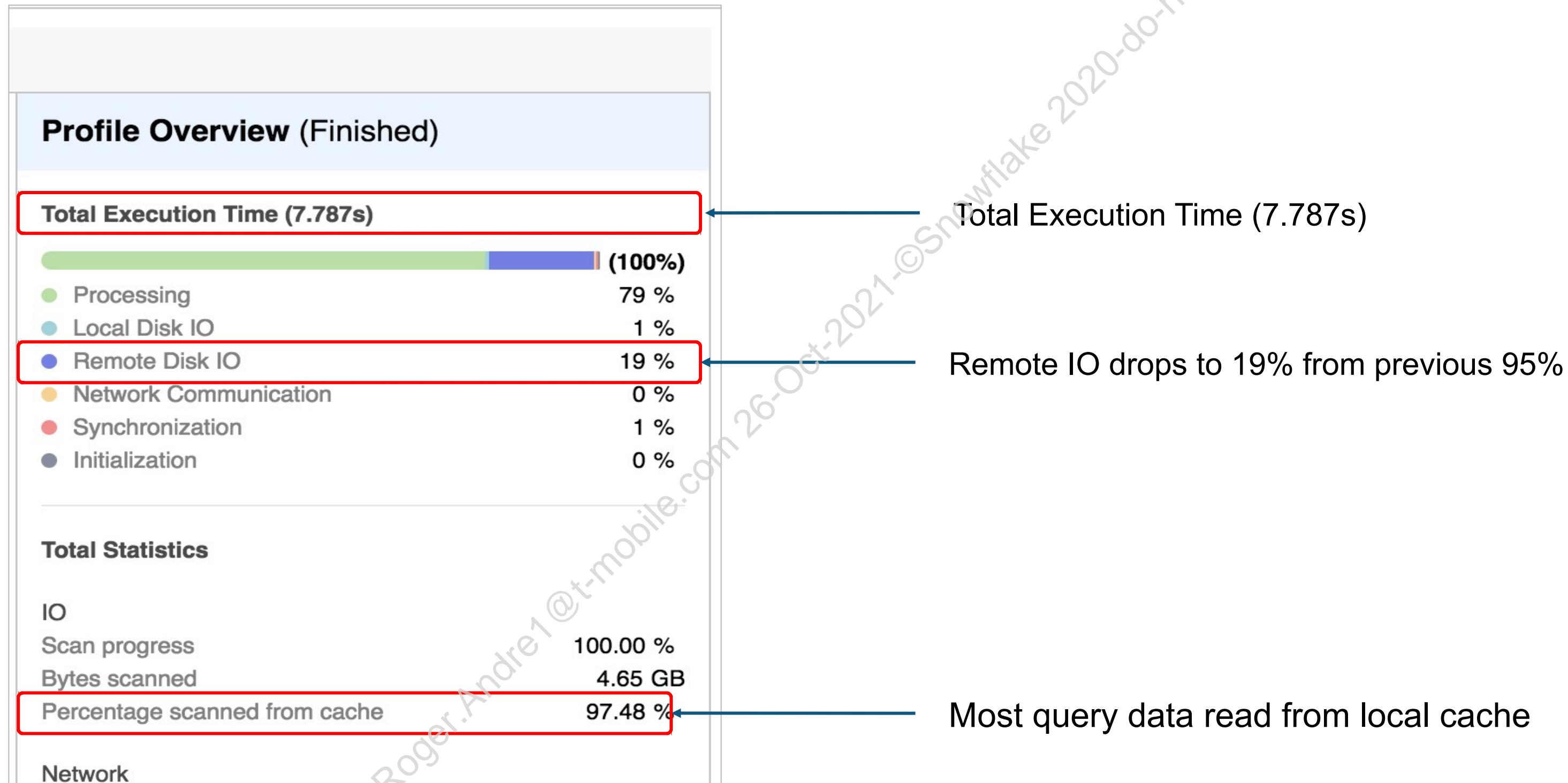
- When a query is run, file headers and column data retrieved are stored on SSD
- Virtual warehouse will first read any locally available data, then read remainder from remote cloud storage
- Data is flushed out in a Least Recently Used (LRU) fashion when cache fills



DATA CACHE: COLD EXAMPLE



DATA CACHE: WARM EXAMPLE



SUMMARY OF CACHE OPTIONS

	Metadata Cache	Query Result Cache	Data Cache
Where is it stored?	Cloud services layer	Cloud services layer and storage layer	Warehouse SSD
What does it store?	Metadata and statistics for micro-partitions and tables	Results set for each query	Data used in query
When is it used?	The optimizer uses this for commands like MIN, MAX, COUNT	Identical query is run again Base table data has not changed	Query using some or all of the same data is run Data has not changed
How long does it last?	Continuously updated	24 hours Clock reset after every run	Until the warehouse is suspended.
Who can use it?	Everyone	For SHOW: user in the same role For SELECT: user with SELECT permissions on all tables	Anyone who uses the same warehouse



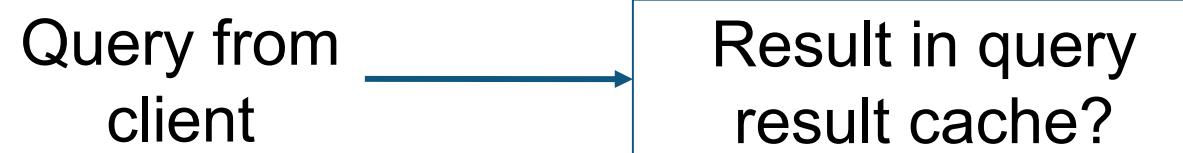
SUMMARY: LIFE CYCLE OF A QUERY

Query from
client



Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy

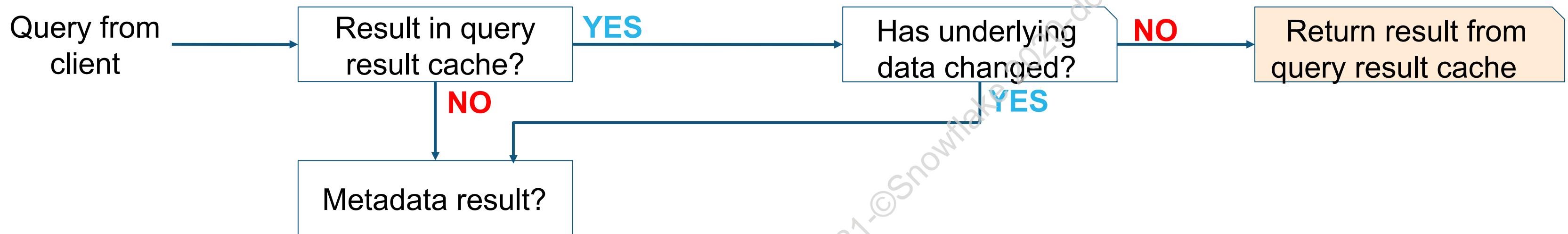
SUMMARY: LIFE CYCLE OF A QUERY



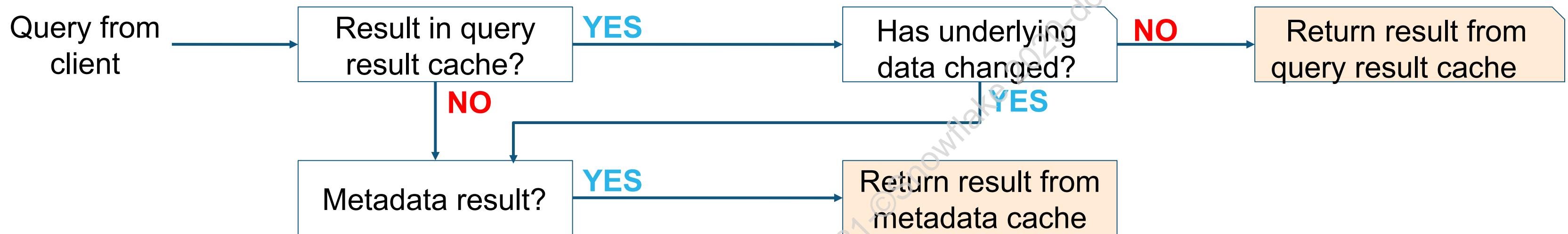
SUMMARY: LIFE CYCLE OF A QUERY



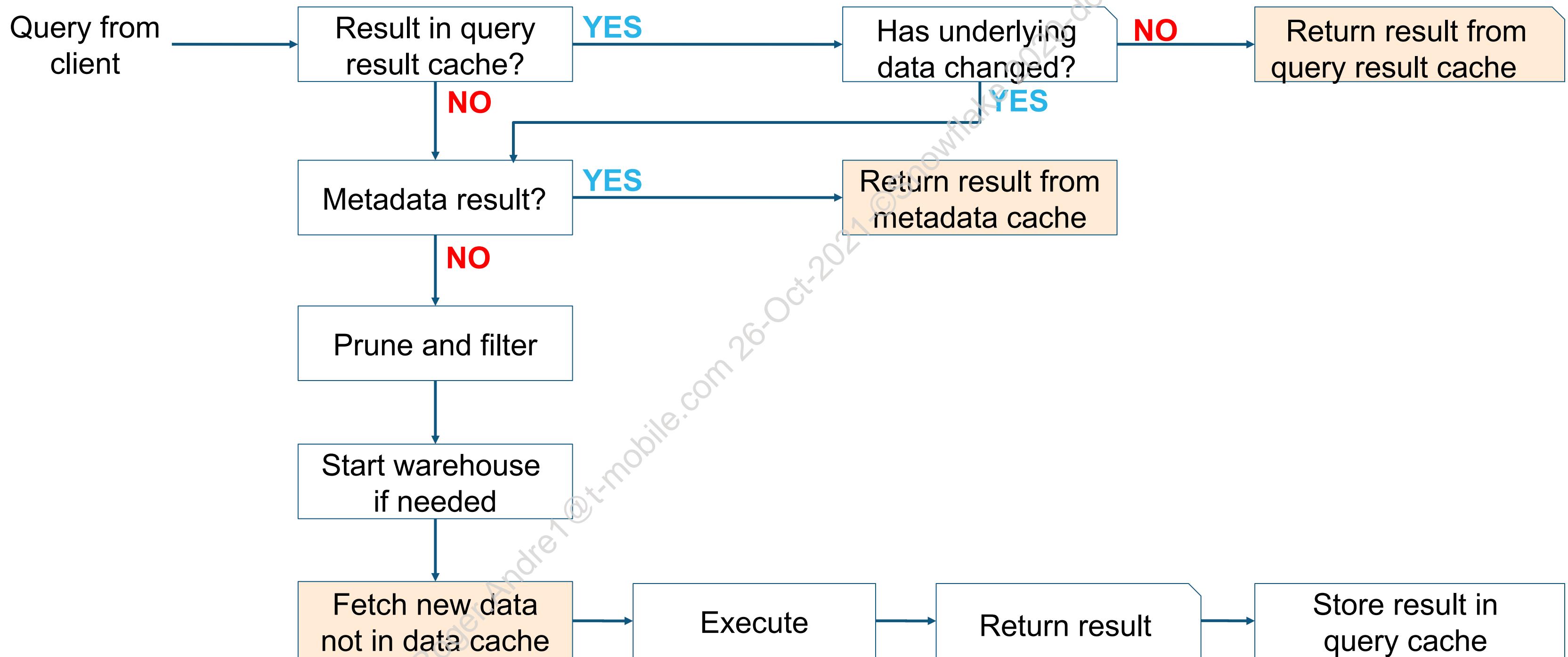
SUMMARY: LIFE CYCLE OF A QUERY



SUMMARY: LIFE CYCLE OF A QUERY



SUMMARY: LIFE CYCLE OF A QUERY



LAB EXERCISE: 4

Explore Snowflake Caching

30 minutes

- Tasks:
- Explore Metadata cache
- Explore Data cache
- Explore Query result cache

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SQL SUPPORT IN SNOWFLAKE

Roger.Andre1@t-mobile.com 26 Oct 2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Querying and Filtering
- Collations
- Subqueries
- Query Tags
- Explain Plan
- Query Profile

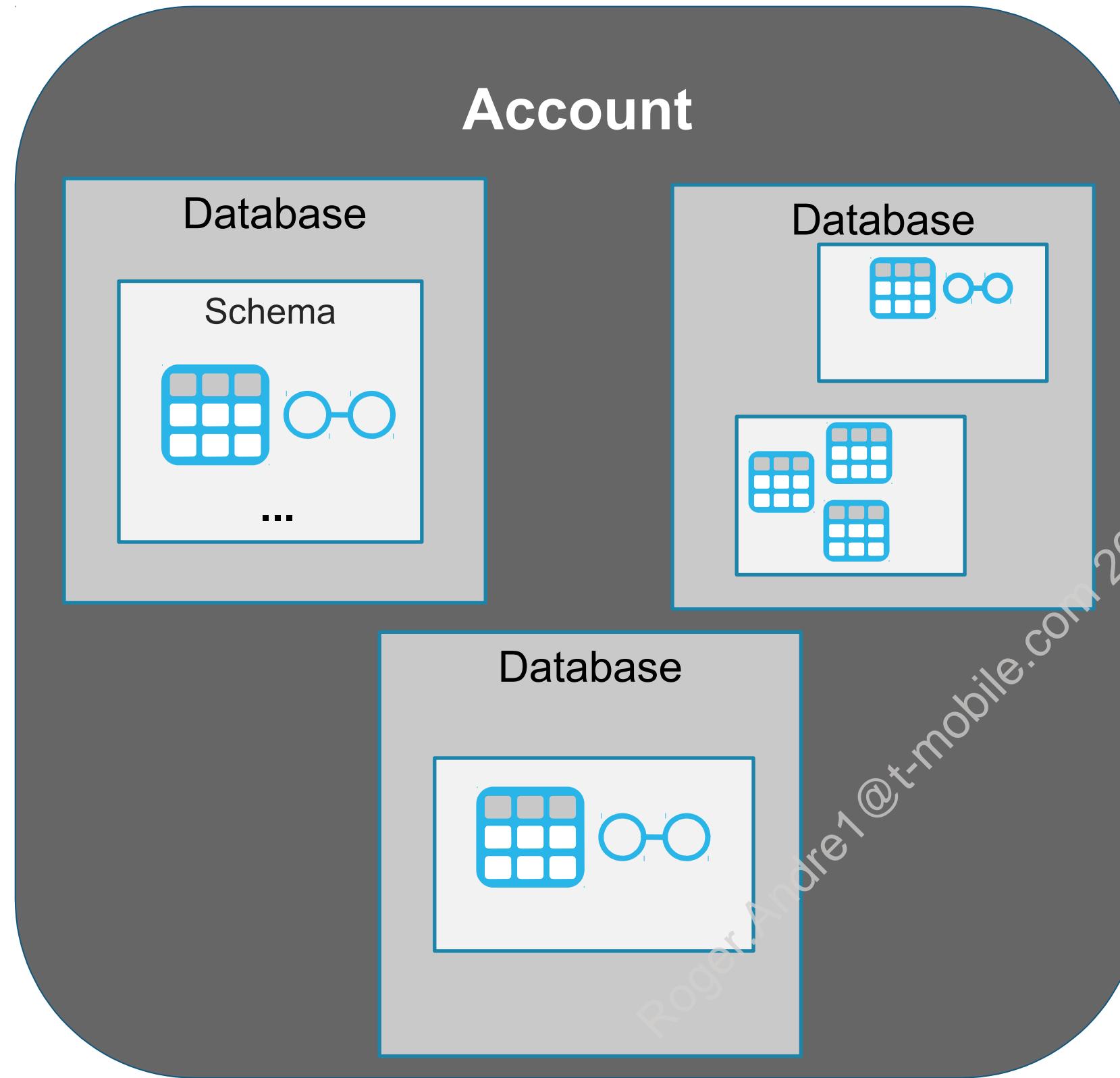
Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SNOWFLAKE STRUCTURE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

LOGICAL DATA ORGANIZATION

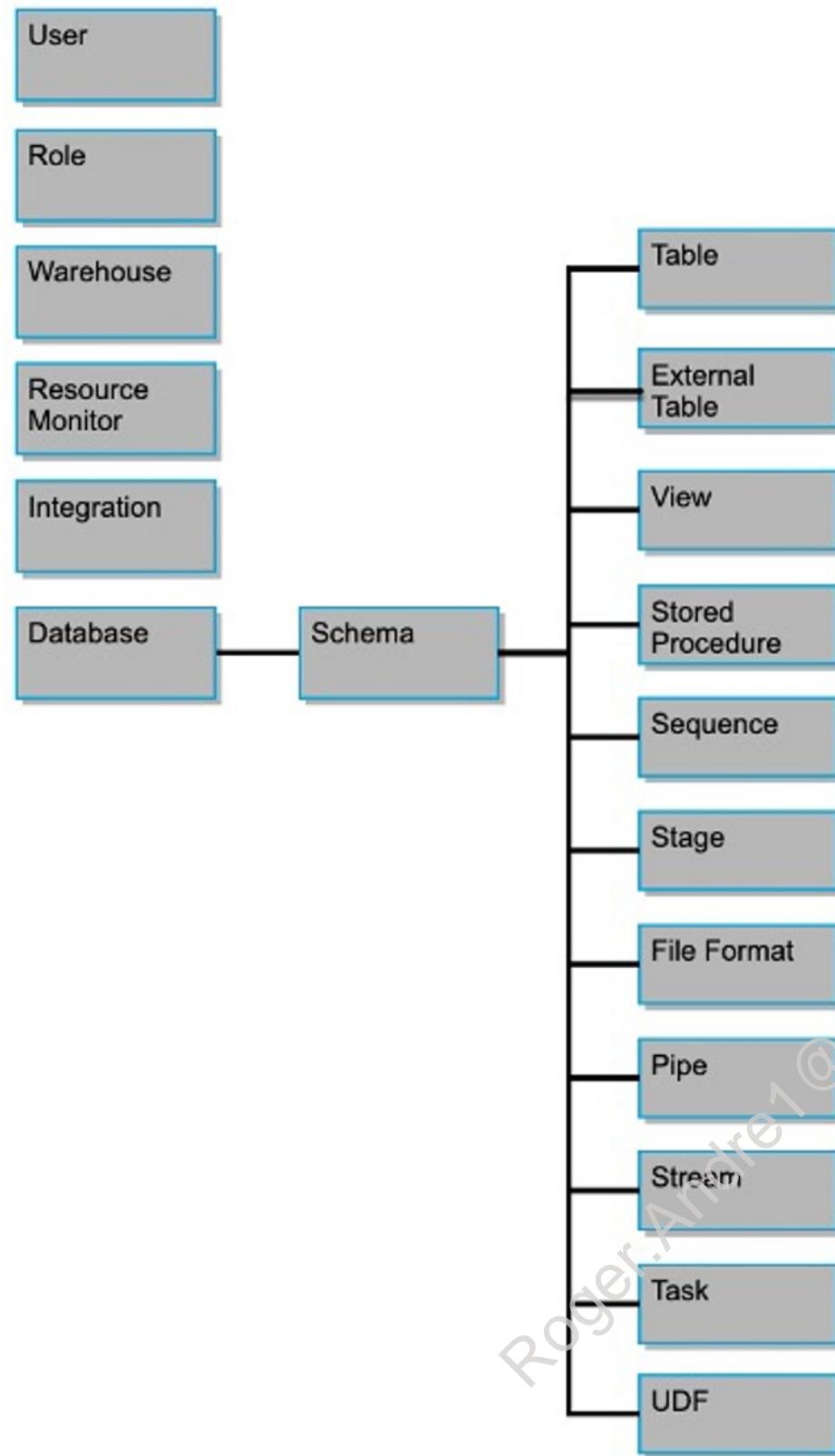


- Databases and schemas logically organize data within a Snowflake account
- A database is a logical grouping of schemas
 - Each database belongs to a single account
- A Schema is a logical grouping of database objects, such as tables and views



SNOWFLAKE OBJECTS

Account



- All Snowflake objects reside within logical containers, with the top level container being the Snowflake Account
- All objects are individually securable
- Users perform operations on objects “privileges” that are “granted” to roles
- Sample Privileges:
 - Create a virtual warehouse
 - List tables in a schema
 - Insert data into a table
 - Select data from a table



TABLE TYPES



PERMANENT

- Persist until dropped
- Designed for data that requires the highest level of data protection and recovery
- Default table type

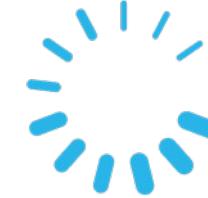
Time Travel

Up to 90 days
with Enterprise

Failsafe



TABLE TYPES



PERMANENT

- Persist until dropped
- Designed for data that requires the highest level of data protection and recovery
- Default table type

TEMPORARY

- Persist and tied to a session (think single user)
- Used for transitory data (for example, ETL/ELT)

Time Travel

Up to 90 days with Enterprise

0 or 1 days

Failsafe



TABLE TYPES



PERMANENT

- Persist until dropped
- Designed for data that requires the highest level of data protection and recovery
- Default table type

TEMPORARY

- Persist and tied to a session (think single user)
- Used for transitory data (for example, ETL/ELT)

TRANSIENT*

- Persist until dropped
- Multiple user
- Used for data that needs to persist, but does not need the same level of data retention as a permanent table

Time Travel

Up to 90 days with Enterprise

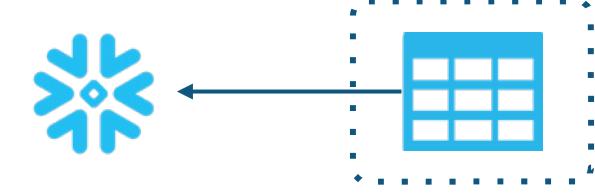
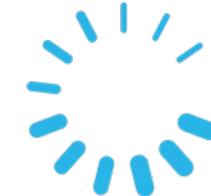
0 or 1 days

0 or 1 days

Failsafe



TABLE TYPES



PERMANENT

- Persist until dropped
- Designed for data that requires the highest level of data protection and recovery
- Default table type

TEMPORARY

- Persist and tied to a session (think single user)
- Used for transitory data (for example, ETL/ELT)

TRANSIENT*

- Persist until dropped
- Multiple user
- Used for data that needs to persist, but does not need the same level of data retention as a permanent table

EXTERNAL

- Persist until removed
- Snowflake “over” an external data lake
- Data accessed via an external stage
- Read-only

Time Travel

Up to 90 days with Enterprise

0 or 1 days

0 or 1 days

x

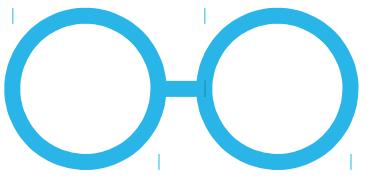
Failsafe



*Transient applicable to Database, Schema and Table



VIEW TYPES

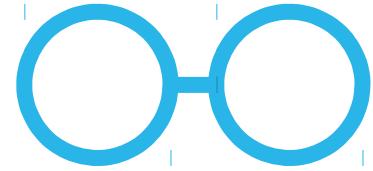


Standard View

- Default view type
- Named definition of a query--
SELECT statement
- Executes as owning role
- Underlying DDL available to any
role with access to the view.



VIEW TYPES



Standard View

- Default view type
- Named definition of a query-- SELECT statement
- Executes as owning role
- Underlying DDL available to any role with access to the view.

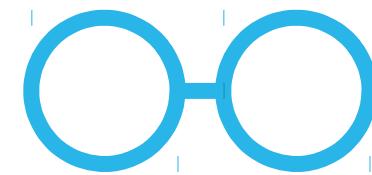


Secure View

- Definition and details only visible to authorized users
- Executes as owning role
- Snowflake query optimizer bypasses optimizations used for regular views

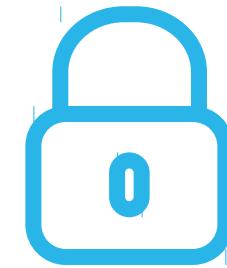


VIEW TYPES



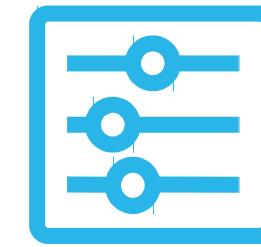
Standard View

- Default view type
- Named definition of a query-- SELECT statement
- Executes as owning role
- Underlying DDL available to any role with access to the view.



Secure View

- Definition and details only visible to authorized users
- Executes as owning role
- Snowflake query optimizer bypasses optimizations used for regular views



Materialized View

- Behaves more like a table
- Results of underlying query are stored
- Auto-refreshed
- Secure Materialized View is also supported



DATA DEFINITION LANGUAGE (DDL)

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MANAGE DATABASE OBJECTS

Database Management

- CREATE/ALTER/DROP/**UNDROP**/USE DATABASE
- SHOW DATABASES

Schema Management

- CREATE/ALTER/DROP/**UNDROP**/USE SCHEMA
- SHOW SCHEMAS

Share Management

- CREATE/ALTER/DROP/DESCRIBE **SHARE**
- SHOW SHARES



CREATE AND MANAGE TABLES

- CREATE TABLE <name> (col1 type, col2 type...)
- CREATE OR REPLACE TABLE
- CREATE TABLE IF NOT EXISTS
- CREATE TABLE ... AS SELECT
- CREATE TABLE ... LIKE (empty copy of existing table)
- CREATE TABLE ... **CLONE**
- ALTER/DROP/**UNDROP** TABLE
- SHOW TABLES
- CREATE TEMPORARY/TRANSIENT/EXTERNAL TABLE



NUMERIC DATA TYPES

Type	Notes
NUMBER, DECIMAL, NUMERIC	Synonymous with NUMBER. Default precision/scale are (38,0)
INT, INTEGER, BIGINT, SMALLINT	Synonymous with NUMBER except precision and scale cannot be specified
DOUBLE, DOUBLE PRECISION, FLOAT, FLOAT4, FLOAT8, REAL	Stored as DOUBLE. A known issue in Snowflake displays these as FLOAT.



STRING, BINARY, AND LOGICAL DATA TYPES

Type	Notes
VARCHAR, STRING, TEXT	Synonymous with VARCHAR. Default (and maximum) is 16,777,216 bytes.
CHAR, CHARACTER	Defaults to VARCHAR(1); maximum is 16,777,216 bytes.
BINARY, VARBINARY	Synonymous with BINARY
BOOLEAN	BOOLEAN



DATE AND TIME DATA TYPES

Type	Notes
DATE	DATE
TIME	TIME
TIMESTAMP	Alias for one of the TIMESTAMP variations, set as a parameter. TIMESTAMP_NTZ by default.
TIMESTAMP_LTZ	TIMESTAMP with local time zone. Time zone, if provided, is not stored.
TIMESTAMP_NTZ, DATETIME	TIMESTAMP with no time zone. Time zone, if provided, is not stored.
TIMESTAMP_TZ	TIMESTAMP with time zone.



OBJECT IDENTIFIER CASE SENSITIVITY

Table Creation Statement

```
CREATE TABLE My_Table
```

Resulting Table Name

MY_TABLE

```
CREATE TABLE my_table
```

MY_TABLE

VS.

Table Create Statement

```
CREATE TABLE "My_Table"
```

Resulting Table Name

My_Table

```
CREATE TABLE "my_table"
```

my_table



CONSTRAINTS

- Snowflake provides support for defining and maintaining constraints
- The only constraint Snowflake enforces is NOT NULL
- Primarily for data modeling purposes, and to support client tools that use constraints
 - Example: Tableau supports using constraints for join culling, which can improve performance



MANAGE VIEWS AND SEQUENCES

- View Management
 - CREATE [OR REPLACE] VIEW
 - CREATE VIEW IF NOT EXISTS
 - CREATE OR REPLACE **SECURE** VIEW
- Materialized View Management
 - CREATE [OR REPLACE] MATERIALIZED VIEW
 - CREATE MATERIALIZED VIEW IF NOT EXISTS
- Sequence Management
 - CREATE [OR REPLACE] SEQUENCE
 - CREATE SEQUENCE IF NOT EXISTS
 - START WITH/INCREMENT BY



DATA MANIPULATION LANGUAGE (DML)

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



STANDARD DML

- Commands for inserting, deleting, updating and merging data
 - INSERT
 - MERGE
 - UPDATE
 - DELETE
 - TRUNCATE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



EXTENDED DML

- File Staging Commands
 - **PUT** (to a stage)
 - **GET** (from a stage)
 - **LIST**
 - **REMOVE**
- Data Loading/Unloading DML
 - **COPY INTO** <table>
 - **COPY INTO** <location>

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUERYING AND FILTERING

Roger.Andre1@t-mobile.com 26-Oct-2021 © Snowflake 2020-do-not-copy



QUERY FORMULATION

Component	What it does
SELECT	Specifies which columns/aggregates/scalar transforms to return
FROM	Defines the data set (table or query) to work with
DISTINCT	Returns only unique values
WHERE	Filters values returned by the FROM clause
LIMIT	Limits the number of records in the returned result set
GROUP BY	Defines how data should be grouped in the results
HAVING	Specifies conditions related to the grouped data
ORDER BY	Specifies how the rows should be ordered
JOIN	Joins multiple tables based on common columns
PIVOT	Rotates a table (turns unique values in a column, into columns)



SELECT

Specifies which columns, aggregates, or scalar transforms to return

Samples:

```
SELECT * FROM region;
```

```
SELECT r_regionkey, r_name, r_comment FROM region;
```

```
SELECT s_phone AS phone_number, CAST(s_acctbal AS DECIMAL(10,2))  
FROM supplier;
```

```
SELECT CURRENT_ROLE();
```

```
SELECT (1+1);
```



FROM

Table, view, or table function to use in a SELECT statement

Samples:

```
SELECT * FROM region;
```

```
SELECT * FROM nation_view;
```

```
SELECT n.n_name as nation_name, r.r_name as region_name  
FROM nation n JOIN region r ON n.n_regionkey = r.r_regionkey;
```

```
SELECT *  
FROM TABLE(INFORMATION_SCHEMA.login_history_by_user());
```



DISTINCT

Returns only unique values

Samples:

```
SELECT DISTINCT (s_nationkey) FROM supplier;
```

```
SELECT COUNT(DISTINCT s_nationkey) FROM supplier;
```



WHERE

Filters the result of the FROM clause, using a *predicate*

Samples:

```
SELECT * FROM invoices WHERE invoice_date < '2018-01-01';
```

```
SELECT * FROM invoices  
WHERE amount < (SELECT AVG(amount) FROM invoices);
```

```
SELECT * FROM invoices  
WHERE invoice_date < DATEADD('DAYS', -30, CURRENT_DATE())  
AND paid=FALSE;
```



LIMIT, FETCH, TOP

ID	NAME	SALARY
1	Joe	53000
2	Rajesh	67000
3	Saira	125000
4	Jenn	98750
...
12390	Anis	45890

```
SELECT * FROM employees  
LIMIT 1;
```

ID	NAME	SALARY
287	June	69340

- Limits the number of rows returned to the value specified
- TOP is used in the column select list instead of later in the query.
 - SELECT TOP 10 * FROM employees;
- The rows returned are non-deterministic unless used with an ORDER BY .



GROUP BY

ITEM_ID	ITEM_TYPE	QTY
1	Pen	57
2	Chair	3
3	Chair	7
4	Table	5
5	Pen	245

```
SELECT item_type, SUM(qty)
FROM inventory
GROUP BY item_type;
```



ITEM_TYPE	QTY
Chair	10
Pen	302
Table	5

- Defines how data should be grouped in the results
- Groups rows that have the same values in a specified column
- Computes aggregate functions for the resulting group if needed



HAVING

Specifies conditions related to grouped data

Samples:

```
SELECT dept FROM employees  
GROUP BY dept HAVING COUNT(*) < 10;
```

```
SELECT item, SUM(qty) FROM orders  
GROUP BY item HAVING SUM(qty) < 100;
```



ORDER BY

MFG	ITEM_TYPE	QTY
OfficeMax	Pen	57
OfficeGuys	Chair	3
Staples	Chair	7
Bob's	Table	5
Staples	Pen	245

```
SELECT item_type, mfg, qty  
FROM inventory ORDER BY item_type;
```

ITEM_TYPE	MFG	QTY
Chair	OfficeGuys	3
Chair	Staples	7
Pen	OfficeMax	57
Pen	Staples	245
Table	Bob's	5

- Orders values in ascending or descending order on a specified column
- Can have multi-level ORDER BY
 - By default, sorts in ascending (ASC) order
 - Use DESC to sort in descending order



JOIN

Combines rows from two tables/views/functions based on common columns

Samples:

```
SELECT n.n_name as nation_name, r.r_name as region_name  
FROM nation n JOIN region r  
ON n.n_regionkey = r.r_regionkey;
```

```
SELECT c_name, o_orderdate FROM customer  
LEFT OUTER JOIN orders ON c_custkey = o_custkey  
ORDER BY o_orderdate DESC;
```



PIVOT

Turns unique values in a column into multiple columns

EmpID	Sales	Month
1	10000	JAN
1	8000	JAN
2	12000	JAN
2	4000	JAN
1	6000	FEB
1	13000	FEB
2	15000	FEB
1	10000	MAR
2	3000	MAR
1	25000	APR
2	31000	APR



PIVOT

Turns unique values in a column into multiple columns

EmplID	Sales	Month
1	10000	JAN
1	8000	JAN
2	12000	JAN
2	4000	JAN
1	6000	FEB
1	13000	FEB
2	15000	FEB
1	10000	MAR
2	3000	MAR
1	25000	APR
2	31000	APR



```
SELECT * FROM monthly_sales  
PIVOT (SUM(sales) for month  
IN ('JAN', 'FEB', 'MAR', 'APR'))  
ORDER BY empid;
```

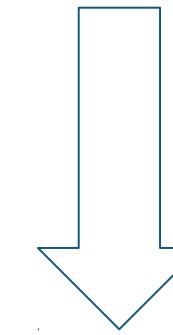
PIVOT

Turns unique values in a column into multiple columns

EmplID	Sales	Month
1	10000	JAN
1	8000	JAN
2	12000	JAN
2	4000	JAN
1	6000	FEB
1	13000	FEB
2	15000	FEB
1	10000	MAR
2	3000	MAR
1	25000	APR
2	31000	APR



```
SELECT * FROM monthly_sales  
PIVOT (SUM(sales) for month  
IN ('JAN', 'FEB', 'MAR', 'APR'))  
ORDER BY empid;
```



EmplID	'JAN'	'FEB'	'MAR'	'APR'
1	18000	19000	10000	25000
2	16000	15000	3000	31000



PIVOT

Turns unique values in a column into multiple columns

```
SELECT * FROM monthly_sales
  PIVOT (SUM(sales)          -- cell values
        FOR month IN ('JAN', 'FEB', 'MAR', 'APR') ) -- columns
ORDER BY empid;
```



COLLATIONS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



COLLATION

- Collate: to collect and combine in proper order
 - Proper order is open to interpretation
- With Snowflake, strings are stored internally in UTF-8
 - UTF-8 uses the numeric representation of characters to sort
 - Upper-case letter have a lower numeric representation than lower-case
 - Example:

Anne

Bob

Jerry

charles

deborah

klaus



SUPPORTED COLLATIONS

SPECIFY HOW TEXT SHOULD BE SORTED/COMPARED

Collation	Use	Examples
Language locale	Language- and country-specific rules to apply	en, en_US, fr, fr_CA, etc.
Case sensitivity	Whether to consider case when comparing values	cs or ci
Accent sensitivity	Whether to consider accented characters equal to, or different from, their base characters	as or ai
Punctuation sensitivity	Whether non-letter characters matter	ps or pi
First letter preference	Whether to sort upper-case or lower-case letters first	f1 or fu
Case conversion	Convert to upper or lower case before comparisons	upper or lower
Space trimming	Remove leading spaces, trailing spaces, or both	trim, ltrim, rtrim



SPECIFYING COLLATIONS ON COLUMNS

```
CREATE TABLE test (col1 VARCHAR, col2 VARCHAR collate 'fr', col3 VARCHAR collate 'es');  
  
INSERT INTO test VALUES  
('Apple', 'Apple', 'Apple'),  
('apple', 'apple', 'apple'),  
('AB CD', 'AB CD', 'AB CD'),  
('ABC', 'ABC', 'ABC'),  
('pino', 'pino', 'pino'),  
('piñata', 'piñata', 'piñata'),  
('ab\'cd', 'ab\'cd', 'ab\'cd'),  
('abc', 'abc', 'abc');  
  
SELECT col1 FROM test;
```

COL1
Apple
apple
AB CD
ABC
pino
piñata
ab'cd
abc



SPECIFYING COLLATIONS ON COLUMNS

```
CREATE TABLE test (col1 VARCHAR, col2 VARCHAR collate 'fr', col3 VARCHAR collate 'es');
```

```
INSERT INTO test VALUES
```

```
('Apple', 'Apple', 'Apple'),  
('apple', 'apple', 'apple'),  
('AB CD', 'AB CD', 'AB CD'),  
('ABC', 'ABC', 'ABC'),  
('pino', 'pino', 'pino'),  
('piñata', 'piñata', 'piñata'),  
('ab\'cd', 'ab\'cd', 'ab\'cd'),  
('abc', 'abc', 'abc');
```

```
SELECT col1 FROM test ORDER BY col1;
```

COL1
Apple
apple
AB CD
ABC
ab'cd
abc
pino
piñata
ab'cd
abc

sorted by
utf8

COL1
AB CD
ABC
Apple
ab'cd
abc
apple
pino
piñata



SPECIFYING COLLATIONS ON COLUMNS

```
CREATE TABLE test (col1 VARCHAR, col2 VARCHAR collate 'fr', col3 VARCHAR collate 'es');
```

```
INSERT INTO test VALUES
```

```
('Apple', 'Apple', 'Apple'),  
('apple', 'apple', 'apple'),  
('AB CD', 'AB CD', 'AB CD'),  
('ABC', 'ABC', 'ABC'),  
('pino', 'pino', 'pino'),  
('piñata', 'piñata', 'piñata'),  
('ab\'cd', 'ab\'cd', 'ab\'cd'),  
('abc', 'abc', 'abc');
```

```
SELECT col1 FROM test ORDER BY col1;
```

```
SELECT col1 FROM test ORDER BY col2;
```

unsorted	sorted by utf8	sorted by french
COL1	COL1	COL1
Apple	AB CD	AB CD
apple	ABC	ab'cd
AB CD	Apple	abc
ABC	ab'cd	ABC
pino	abc	apple
piñata	apple	Apple
ab'cd	pino	piñata
abc	pino	piñata



SPECIFYING COLLATIONS ON COLUMNS

```
CREATE TABLE test (col1 VARCHAR, col2 VARCHAR collate 'fr', col3 VARCHAR collate 'es');
```

```
INSERT INTO test VALUES
```

```
('Apple', 'Apple', 'Apple'),  
('apple', 'apple', 'apple'),  
('AB CD', 'AB CD', 'AB CD'),  
('ABC', 'ABC', 'ABC'),  
('pino', 'pino', 'pino'),  
('piñata', 'piñata', 'piñata'),  
('ab\'cd', 'ab\'cd', 'ab\'cd'),  
('abc', 'abc', 'abc');
```

```
SELECT col1 FROM test ORDER BY col1;
```

```
SELECT col1 FROM test ORDER BY col2;
```

```
SELECT col1 FROM test ORDER BY col3;
```

unsorted	sorted by utf8	sorted by french	sorted by spanish
COL1	COL1	COL1	COL1
Apple	AB CD	AB CD	AB CD
apple	ABC	ab'cd	ab'cd
AB CD	Apple	abc	abc
ABC	ab'cd	ABC	ABC
pino	abc	apple	apple
piñata	apple	Apple	Apple
ab'cd	pino	piñata	pino
abc	piñata	pino	piñata



SPECIFYING COLLATIONS IN COMPARISONS

```
CREATE TABLE test (c1 VARCHAR, c2 VARCHAR);
```

```
INSERT INTO test VALUES
('its', 'it\s'),
('log in', 'login'),
('MyTable', 'mytable');
```



C1	C2
its	it's
log in	login
MyTable	mytable



SPECIFYING COLLATIONS IN COMPARISONS

```
CREATE TABLE test (c1 VARCHAR, c2 VARCHAR);
```

```
INSERT INTO test VALUES  
('its', 'it\'s'),  
('log in', 'login'),  
('MyTable', 'mytable');
```

```
SELECT * FROM test  
WHERE c1=c2;
```



C1	C2
its	it's
log in	login
MyTable	mytable

C1	C2



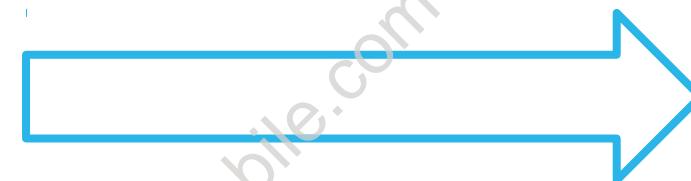
SPECIFYING COLLATIONS IN COMPARISONS

```
CREATE TABLE test (c1 VARCHAR, c2 VARCHAR);
```

```
INSERT INTO test VALUES
('its', 'it\'s'),
('log in', 'login'),
('MyTable', 'mytable');
```

```
SELECT * FROM test
WHERE c1=c2;
```

```
SELECT * FROM test
WHERE c1=c2 collate 'en-ci';
```



C1	C2
its	it's
log in	login
MyTable	mytable

C1	C2

C1	C2
MyTable	mytable



SPECIFYING COLLATIONS IN COMPARISONS

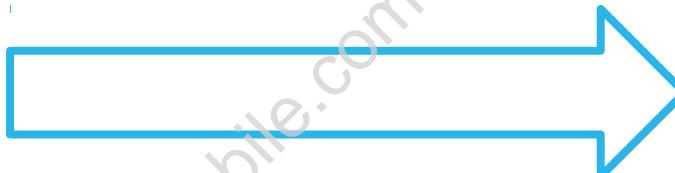
```
CREATE TABLE test (c1 VARCHAR, c2 VARCHAR);
```

```
INSERT INTO test VALUES  
('its', 'it\'s'),  
('log in', 'login'),  
('MyTable', 'mytable');
```

```
SELECT * FROM test  
WHERE c1=c2;
```

```
SELECT * FROM test  
WHERE c1=c2 collate 'en-ci';
```

```
SELECT * FROM test  
WHERE c1=c2 collate 'en-ci-pi';
```



C1	C2
its	it's
log in	login
MyTable	mytable

C1	C2

C1	C2
MyTable	mytable

C1	C2
its	it's
log in	login
MyTable	mytable



FUNCTIONS THAT SUPPORT COLLATION

- [NOT] BETWEEN
- CASE
- COALESCE
- CONCAT, ||
- CONTAINS
- DECODE
- ENDSWITH
- EQUAL_NULL
- GET_DDL
- GREATEST
- IFF
- IFNULL
- LEAST
- LEFT
- LISTAGG
- LPAD
- MIN / MAX
- NULLIF
- NVL
- NVL2
- RIGHT
- RPAD
- STARTSWITH



SUBQUERIES

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SUBQUERY OVERVIEW

- A Subquery is a query within another query

Outer/main query
Generally executed 2nd

```
SELECT <columns>  
FROM <table>  
WHERE column_name <expression> <operator>
```

Subquery
Generally executed 1st

```
( SELECT <columns>  
FROM <table>  
WHERE ...  
);
```



ADDITIONAL SUBQUERY INFO

- Can be in WHERE, HAVING, and FROM clauses
- Can be used with SELECT, UPDATE, INSERT, DELETE along with an expression operation
- Subqueries are on the right side of the comparison operator:

```
SELECT rep, total_sales  
FROM sales_results  
WHERE total_sales >  
(SELECT AVG(total_sales)  
FROM sales_results)
```



USE TEMP TABLE FOR REPETITIVE SUBQUERIES

If you're using the same subquery multiple times, use a temporary table to materialize subquery results:

- Improves performance by reducing repeated I/O
- Saves cost by reducing repeated computation
- Temporary table exists only within session

```
create temporary table  
mydb.public.customer_total_return as (  
    select sr_customer_sk as ctr_customer_sk,  
          sr_store_sk as ctr_store_sk,  
          sum(SR_RETURN_AMT_INC_TAX) as ctr_total_return  
     from store_returns  
   ,date_dim  
    where sr_returned_date_sk = d_date_sk  
      and d_year =1999  
  group by sr_customer_sk,sr_store_sk  
);  
  
select c_customer_id  
  from mydb.public.customer_total_return ctr1  
 ,store  
 ,customer  
 where ctr1.ctr_total_return >  
       (select avg(ctr_total_return)*1.2  
        from mydb.public.customer_total_return ctr2  
       where ctr1.ctr_store_sk = ctr2.ctr_store_sk  
     )  
   and s_store_sk = ctr1.ctr_store_sk  
   and s_state = 'NM'  
   and ctr1.ctr_customer_sk = c_customer_sk  
  order by c_customer_id  
 limit 100;
```



QUERY TAGS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



USE QUERY TAGS

1. Set up a query tag

- `ALTER SESSION SET QUERY_TAG = '<string>'`

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



USE QUERY TAGS

1. Set up a query tag

- ALTER SESSION SET QUERY_TAG = '<string>'

2. Filter History by query tag:

The screenshot shows the Snowflake Query History interface. At the top, there is a search bar with the placeholder "Display queries that meet all of the following criteria:" and a dropdown menu set to "Query Tag". Below the search bar are two checkboxes: "Include client-generated statements" and "Include queries executed by user tasks". To the right of the search bar is a "Clear filters" button. A large blue oval highlights the search bar and the checkboxes. A blue arrow points from the search bar area down to the "Query Tag" column in the results table. Another blue circle highlights the "Query Tag" column header. The results table has columns: Session ID, Start Time, End Time, Total Duration, Bytes Scanned, Client Info, Rows, and Query Tag. There are three rows of data, each with a "TagTest" value in the "Query Tag" column.

	Session ID	Start Time	End Time	Total Duration	Bytes Scanned	Client Info	Rows	Query Tag
II	5554681580...	4:38:16 PM	4:38:17 PM	456ms	4KB	Snowflake UI 202006...	25	TagTest
II	5554681580...	4:38:07 PM	4:38:10 PM	3.0s	40.3MB	Snowflake UI 202006...	1.5M	TagTest
II	5554681580	4:38:26 PM	4:38:27 PM	1.6s	10.2MB	Snowflake UI 202006	25	TagTest



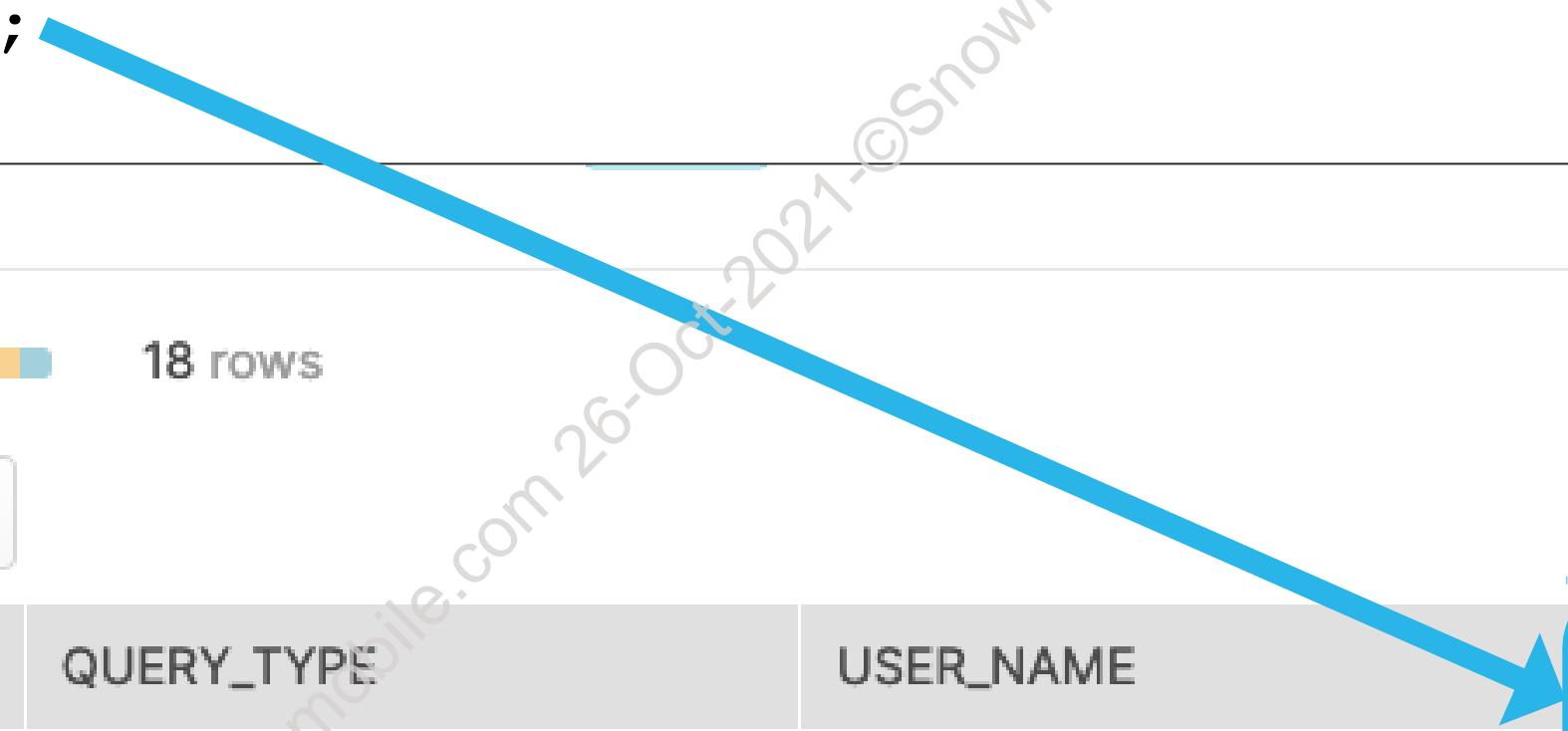
LOCATE TAGGED QUERIES

```
SELECT query_id, query_type, user_name, query_tag  
FROM SNOWFLAKE.ACOUNT_USAGE.QUERY_HISTORY  
WHERE start_time > '2020-05-01'  
AND query_tag = 'TagTest';
```

Results Data Preview ← Open History

✓ Query ID SQL 8.91s  18 rows

Filter result...   Columns ▾



Row	QUERY_ID	QUERY_TYPE	USER_NAME	QUERY_TAG
1	0195216e-0291-4032-000...	SELECT	DLITTLEFIELD_SFC	TagTest
2	0195216e-02e2-1055-0000...	SELECT	DLITTLEFIELD_SFC	TagTest
3	01952170-02d8-ac1-0000...	SHOW	DLITTLEFIELD_SFC	TagTest
4	01952170-0234-c248-000...	SHOW	DLITTLEFIELD SFC	TagTest



EXPLAIN PLAN

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



WHAT IS “EXPLAIN PLAN”?

EXPLAIN

```
SELECT  
    l_orderkey  
FROM  
    lineitem  
LIMIT 10;
```

- Snowflake command that displays the logical execution steps without executing
- Results show in JSON or tabular format
- Helpful for performance tuning and for controlling costs



EXPLAIN SYNTAX AND EXAMPLES

JSON AND TABULAR

EXPLAIN USING JSON

```
SELECT l_orderkey  
FROM lineitem LIMIT 10;
```

```
{  
  "GlobalStats": {  
    "bytesAssigned": "165228544",  
    "partitionsAssigned": "10",  
    "partitionsTotal": "10"  
  },  
  "Operations": [  
    [  
      {  
        "expressions": [  
          "LINEITEM.L_ORDERKEY"  
        ],  
        "id": 1,  
        "isFinal": true,  
        "operator": "SELECT",  
        "planNodeId": 1,  
        "type": "OperationNode"  
      }  
    ]  
  ]  
}
```

EXPLAIN

```
SELECT l_orderkey  
FROM lineitem LIMIT 10;
```

	partitionsTotal	partitionsAssigned	bytesAssigned
10...	10	10	165228544
0...	NULL	NULL	NULL
0	NULL	NULL	NULL
Y	10	10	165228544



WHAT'S IN A PLAN?

KEY POINTS

- Partition Pruning
- Join Ordering
- Join Types

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



WHAT IS IN A PLAN?

Partition Pruning

Results

row#	Total Partitions	Scanned Partitions
1	"86546"	"10584"



WHAT IS IN A PLAN?

Join Ordering

parent	operation	expressions
NULL	GlobalStats	NULL
NULL	Result	LINEITEM.L_ORDERKEY, SUM(LINEITEM.L_EXTENDEDPRICE * (1 - LINEITEM.L_DISCOUNT)) AS L_EXTENDEDPRICE, SUM(LINEITEM.L_EXTENDEDPRICE * (1 - LINEITEM.L_DISCOUNT)) * LINEITEM.L_TAXES AS L_EXTENDEDPRICE_TAXES, SUM(LINEITEM.L_EXTENDEDPRICE * (1 - LINEITEM.L_DISCOUNT)) * (LINEITEM.L_TAXES + LINEITEM.L_DISCOUNT) AS L_EXTENDEDPRICE_DISCOUNT_TAXES
0	SortWithLimit	sortKey: [SUM(LINEITEM.L_EXTENDEDPRICE * (1 - LINEITEM.L_DISCOUNT))]
1	Aggregate	aggExprs: [SUM(LINEITEM.L_EXTENDEDPRICE * (1 - LINEITEM.L_DISCOUNT))]
2	InnerJoin	joinKey: (ORDERS.O_ORDERKEY = LINEITEM.L_ORDERKEY)
3	InnerJoin	joinKey: (CUSTOMER.C_CUSTKEY = ORDERS.O_CUSTKEY)



WHAT IS IN A PLAN?

Join Types

operation	objects	expressions
GlobalStats	NULL	NULL
Result	NULL	D.D_DATE, SUM((S.SS_QUANTITY) * (S.SS...
Aggregate	NULL	aggExprs: [SUM((S.SS_QUANTITY) * (S.SS...
CartesianJoin	NULL	NULL
TableScan	SNOWFLAKE_SAMPLE_DATA.TPCDS_SF10...	D_DATE
TableScan	SNOWFLAKE_SAMPLE_DATA.TPCDS_SF10...	SS_QUANTITY, SS_SALES_PRICE



TOP USE CASE

VALIDATING PREDICATE (WHERE) PARTITION PRUNING

**154 of 3242
PARTITIONS
GOOD PREDICATE!**

```
SELECT
    count(*)
FROM
    ORDERS
WHERE
    o_orderdate >= to_date('1998-01-01');
```

operation	objects	alias	expressions	partitionsTotal	partitionsAssign
GlobalStats	NULL	NULL	NULL	3242	154
Result	NULL	NULL	COUNT(*)	NULL	NULL
Aggregate	NULL	NULL	aggExprs: [C...	NULL	NULL
Filter	NULL	NULL	ORDERS.O...	NULL	NULL
TableScan	SNOWFLAK...	NULL	O_ORDERDA...	3242	154



QUERY PROFILE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUERY STATISTICS - HISTORY TAB

- Provides a tabular, high-level view of each query
- Includes basic performance metrics (duration, bytes scanned, rows)
- Color-coded to provide quick insights

Hide Filters View SQL Abort...

Display queries that meet all of the following criteria:

User is DGARDNER

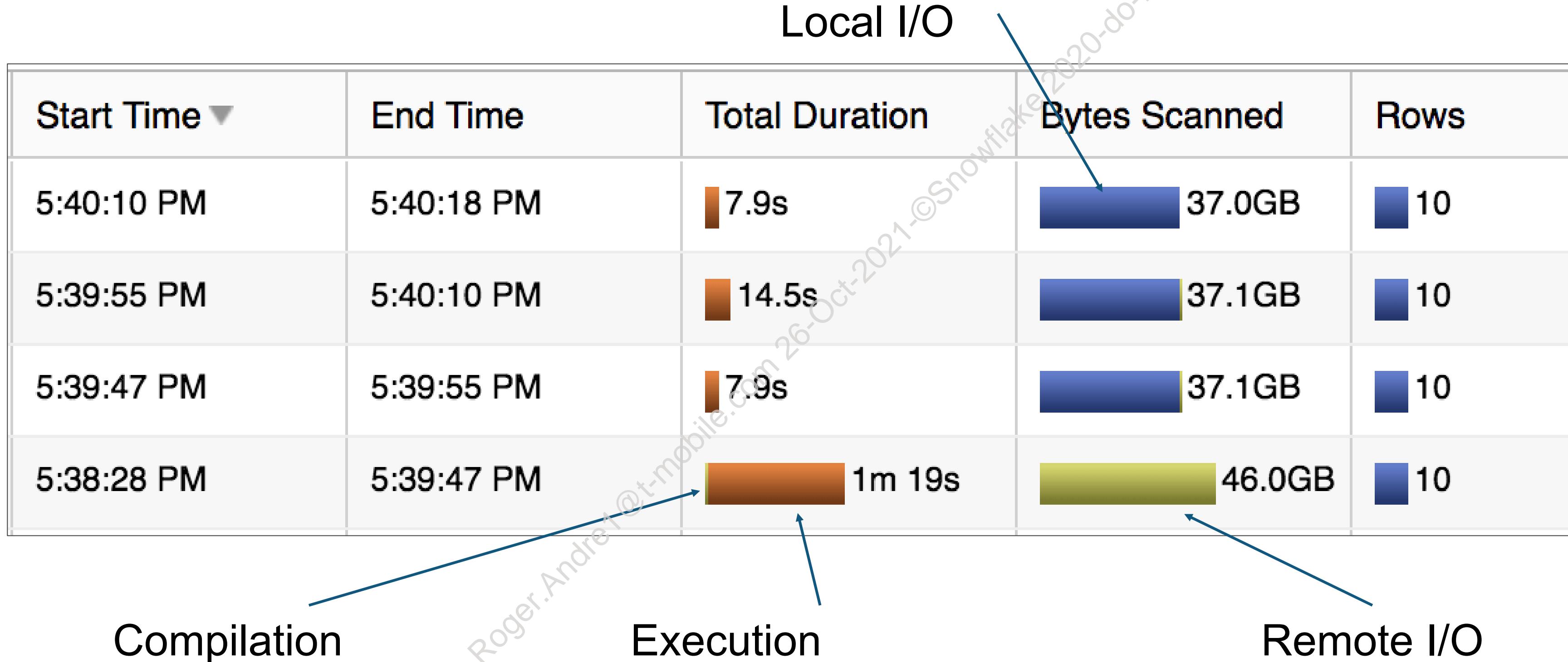
Include client-generated statements

Clear filters

Status	Query ID	SQL Text	User	Warehouse	Clust...	Size	Session ID	Start Time	End Time	Total Duration	Bytes Scanned	Rows	Query Tag
✓	8b92ffbf-73...	ALTER WAREHOUSE PS_BOOTCAMP_WH SET WAR...	DGARDNER	PS_BOOTC...			7384337285...	5:40:18 PM	5:40:18 PM	119ms			
✓	1d912b31-...	ALTER WAREHOUSE PS_BOOTCAMP_WH SUSPEND ;	DGARDNER	PS_BOOTC...			7384337285...	5:40:18 PM	5:40:18 PM	108ms			
✓	6c8b59d5-...	ALTER SESSION SET USE_CACHED_RESULT = FALS...	DGARDNER	PS_BOOTC...			7384337285...	5:40:18 PM	5:40:18 PM	36ms			
✓	0c3ada06-...	WITH X AS (SELECT SS.SS_STORE_SK ,SUM(SS.SS...	DGARDNER	PS_BOOTC...	1	X-Large	7384337285...	5:40:10 PM	5:40:18 PM	7.9s	37.0GB	10	
✓	9259f513-a...	WITH X AS (SELECT SS.SS_STORE_SK ,SUM(SS.SS...	DGARDNER	PS_BOOTC...	1	X-Large	7384337285...	5:39:55 PM	5:40:10 PM	14.5s	37.1GB	10	
✓	8fe8a02e-6...	ALTER SESSION SET USE_CACHED_RESULT = TRUE ;	DGARDNER	PS_BOOTC...			7384337285...	5:39:55 PM	5:39:55 PM	46ms			
✓	ae9eb44b-...	WITH X AS (SELECT SS.SS_STORE_SK ,SUM(SS.SS...	DGARDNER	PS_BOOTC...	1	X-Large	7384337285...	5:39:47 PM	5:39:55 PM	7.9s	37.1GB	10	
✓	4866fbf0-8...	WITH X AS (SELECT SS.SS_STORE_SK ,SUM(SS.SS...	DGARDNER	PS_BOOTC...	1	X-Large	7384337285...	5:38:28 PM	5:39:47 PM	1m 19s	46.0GB	10	
✓	dd1182cd...	ALTER WAREHOUSE PS_BOOTCAMP_WH RESUME .	DGARDNER	PS_BOOTC...			7384337285...	5:39:28 PM	5:39:28 PM	174ms			

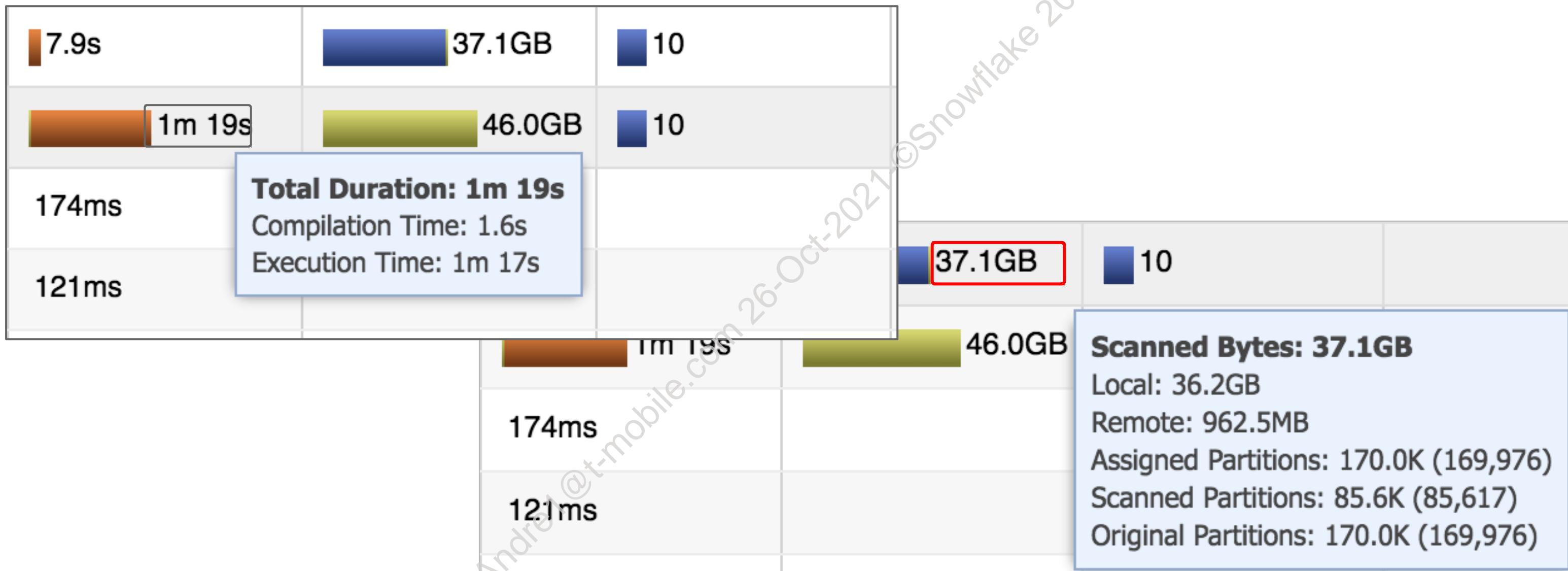


QUERY STATISTICS - HISTORY TAB



QUERY STATISTICS - HISTORY TAB

- Hover over numeric value to see breakdown



ACCESS THE QUERY PROFILE

- Multiple ways to access the SQL Profiler

The screenshot shows the 'Results' tab of a SQL worksheet. The query ID is 34e4e084-... and it took 6.71s to run, returning 10 rows. The results are displayed in a table with columns Row, C_CUSTKEY, C_NAME, and C_ADDRESS. Rows 1, 2, and 3 are visible.

Row	C_CUSTKEY	C_NAME	C_ADDRESS
1	60001	Customer#0000...	9li4zQn9cX
2	60002	Customer#0000...	ThGBMjDwKzko...
3	60003	Customer#0000...	Ed hbPtTXMTAs...

Query results pane
in worksheet

The screenshot shows the 'History' tab of the Snowflake interface. It displays a list of generated statements. One statement, with Query ID db40cb27-..., is highlighted with a red box. A tooltip for this query shows its SQL text: `ALTER WAREHOUSE PS_BOOTCAMP_WH SUSPEND ;`.

Status	Query ID	SQL Text	User	Warehouse	Clus
✓	34e4e084-...	ALTER WAREHOUSE PS_BOOTCAMP_WH SET WAR...	DGARDNER	PS_BOOTC...	
✓	db40cb27-...	ALTER WAREHOUSE PS_BOOTCAMP_WH SUSPEND ;	DGARDNER	PS_BOOTC...	
✓	2685f614-e...	WITH X AS (SELECT SS.SS_STORE_SK ,SUM(SS.SS...	DGARDNER	PS_BOOTC...	1
✓	b932d4ea-	Query ID: 2685f614-e220-4cbf-a5e9-3856eddeddd4c RESUME ;	DGARDNER	PS_BOOTC...	

History tab

QUERY PROFILE - DETAILS TAB

- Clicking on a Query ID brings up this view:

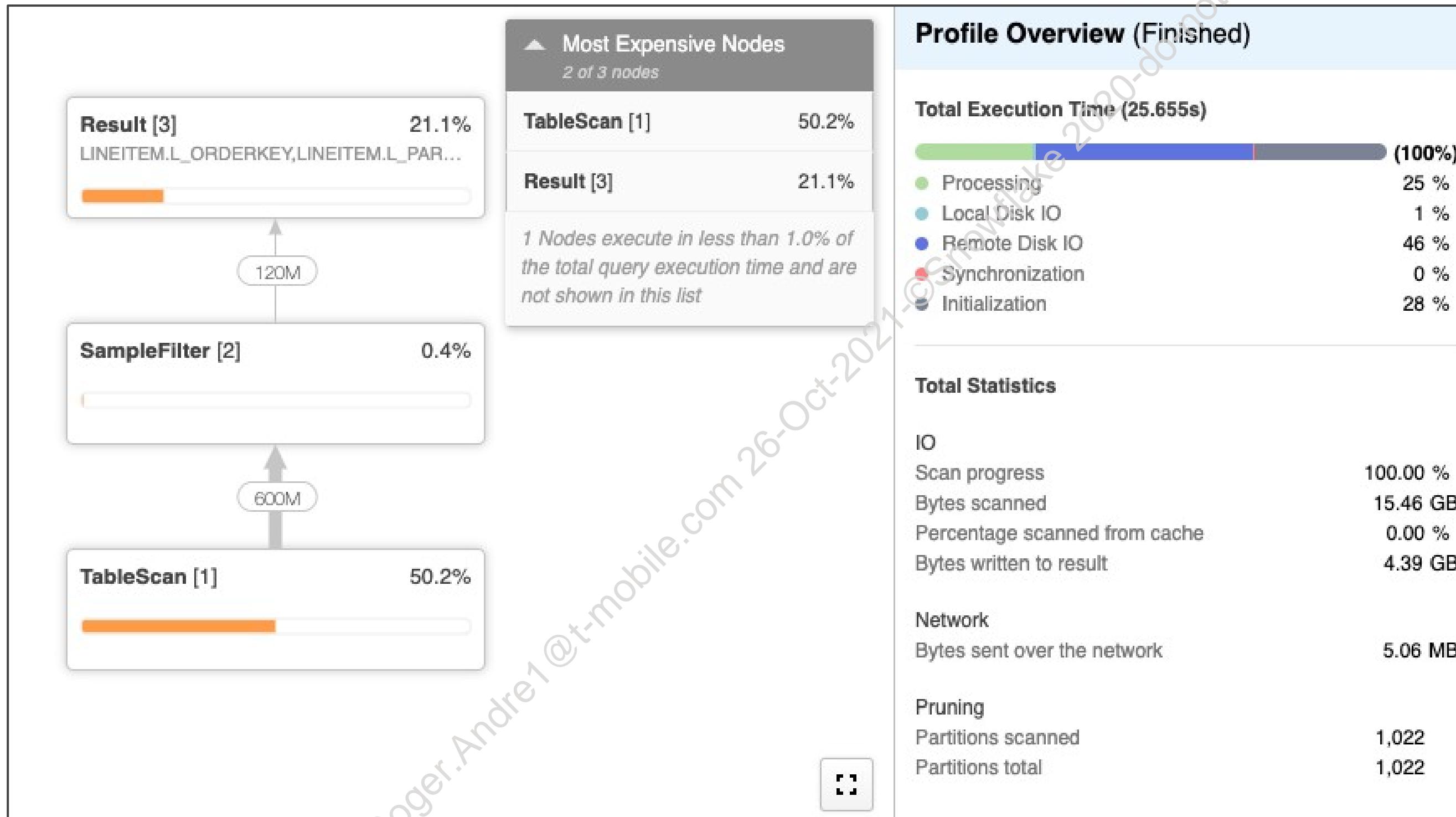
Details		Profile
Status	Success	
User	DGARDNER	
Warehouse	PS_BOOTCAMP_WH	
Start Time	11/15/18 4:32:23 PM	
End Time	11/15/18 4:32:34 PM	
Total Duration	11.4s	
Scanned Bytes	879.3MB	
Rows	10	
Query ID	2685f614-e220-4cbf-a5e9-3856eddeddd4c	
Session ID	4401484	

SQL Text

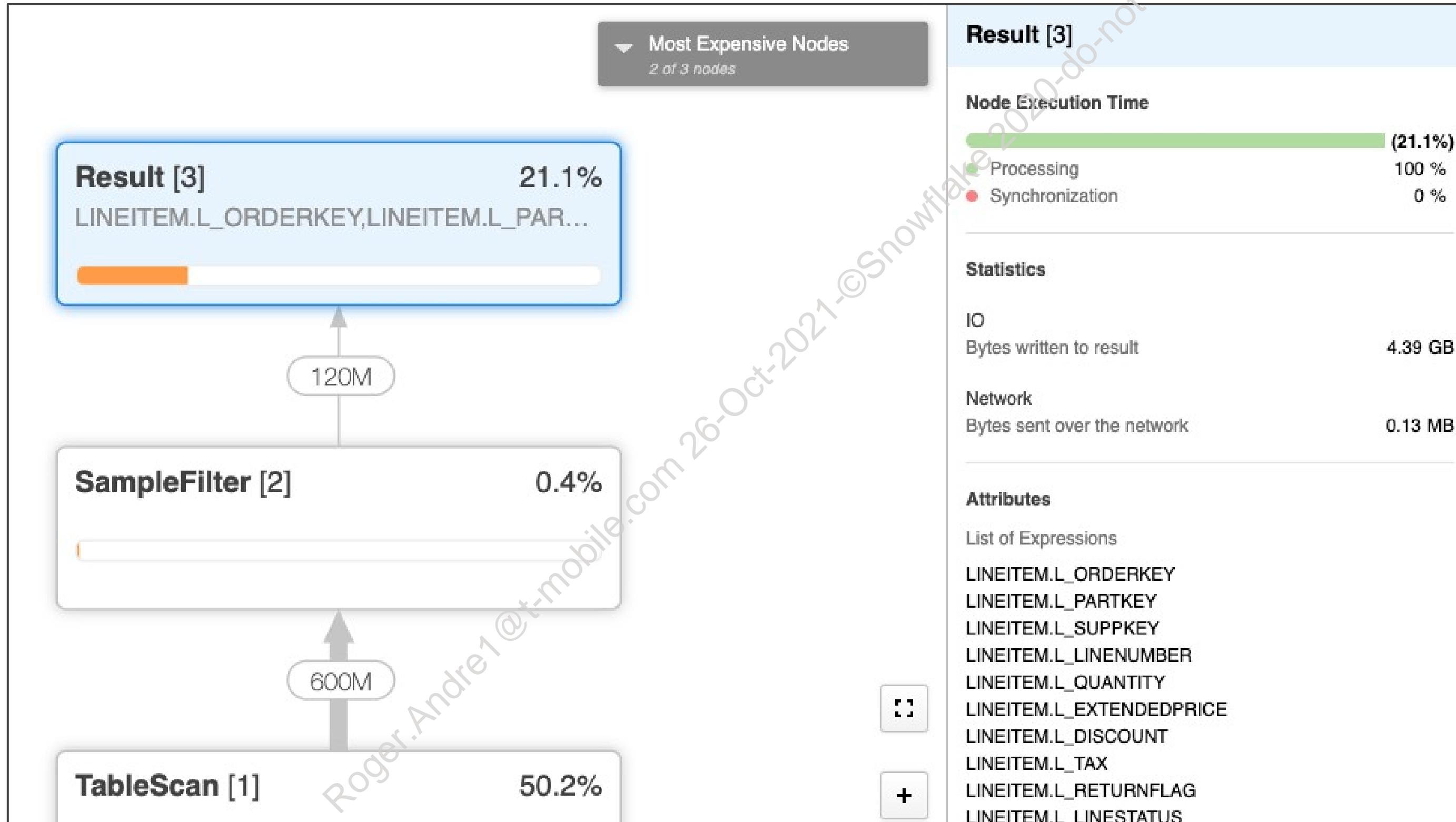
```
1 WITH X AS (
2     SELECT SS.SS_STORE_SK
3         ,SUM(SS.SS_EXT_SALES_PRICE) AS SUM_SS_EXT_SALES_PRICE
4     FROM STORE_SALES_CLUST_DATE_ITEM SS
5         JOIN SAMPLE_DATA.TPCDS_SF10TCL.DATE_DIM DD
6             ON DD.D_DATE_SK = SS.SS SOLD_DATE_SK
7     WHERE DD.D_DATE >= DATEADD(DAY, -6, '2003-01-02' ::DATE)
8         AND DD.D_DATE <= '2003-01-02' ::DATE
9         AND SS.SS_STORE_SK IS NOT NULL
10    GROUP BY SS.SS_STORE_SK
11    ORDER BY SUM_SS_EXT_SALES_PRICE DESC
12    LIMIT 10
13 )
14 SELECT S.S_STORE_NAME
15     ,X.SUM_SS_EXT_SALES_PRICE
16     FROM X
17         JOIN SAMPLE_DATA.TPCDS_SF10TCL.STORE S
18             ON S.S_STORE_SK = X.SS_STORE_SK
19     ORDER BY X.SUM_SS_EXT_SALES_PRICE DESC
20 ;
```



QUERY PROFILE - PROFILE TAB



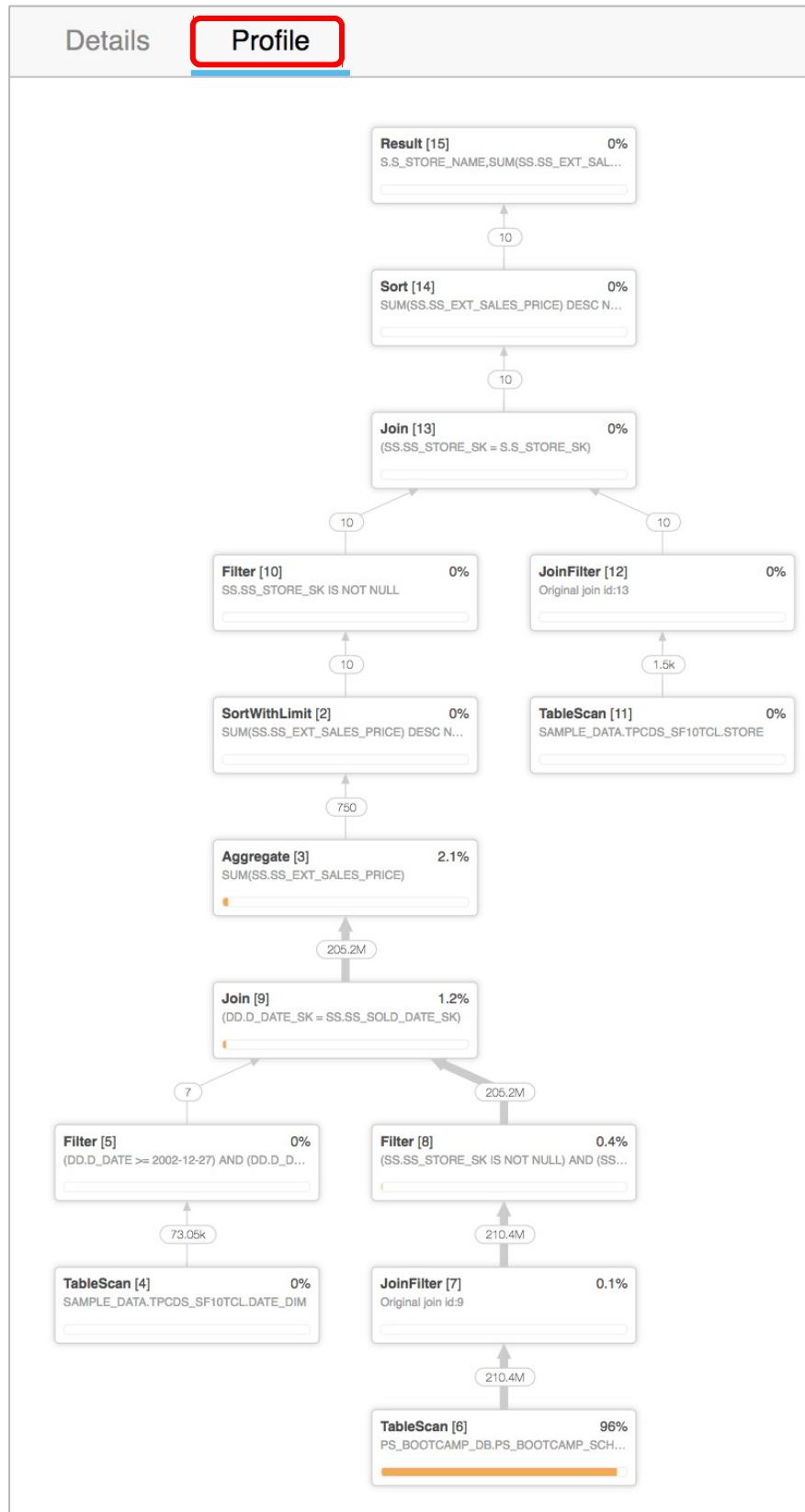
QUERY PROFILE - MORE DETAILS



MULTI-STEP QUERIES



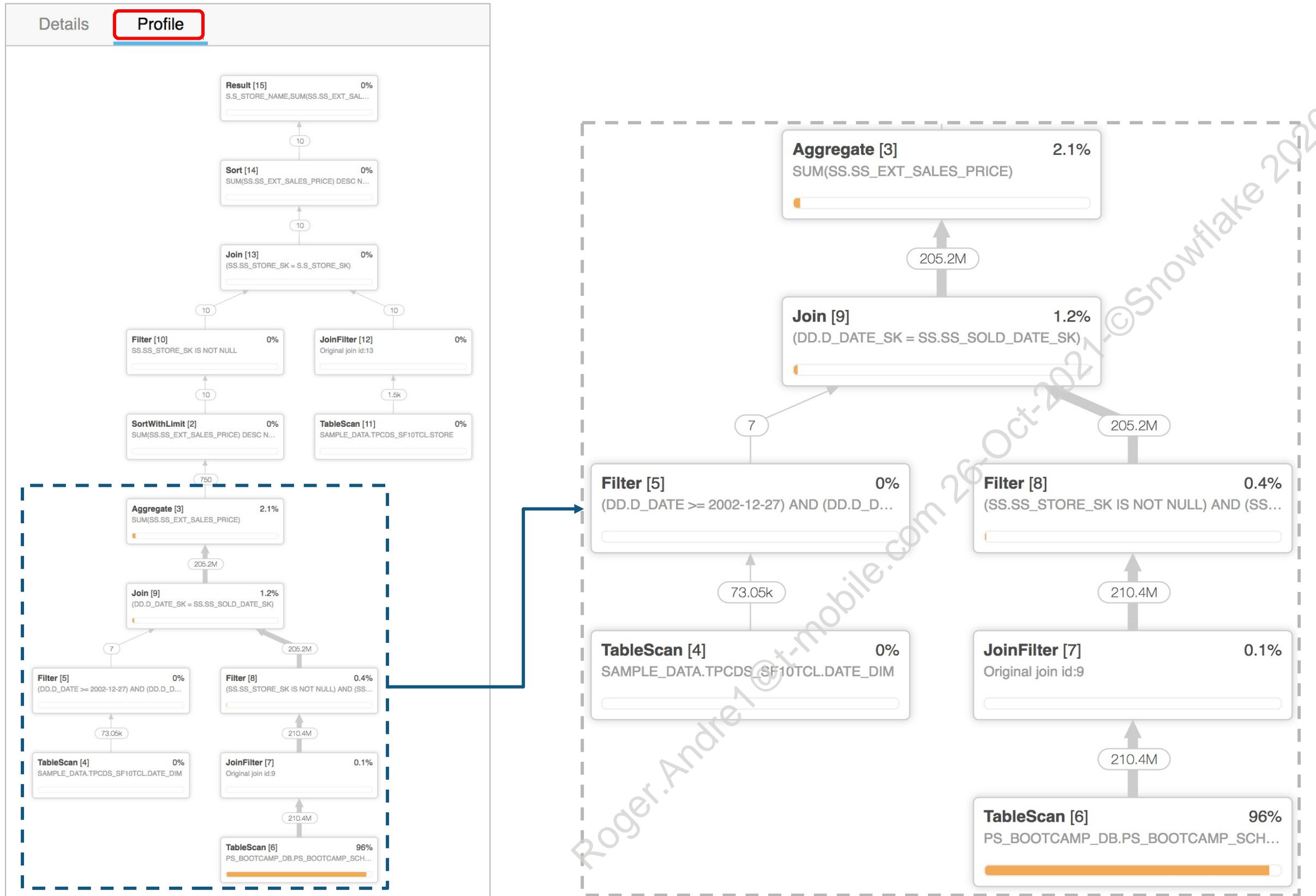
READING A QUERY PROFILE



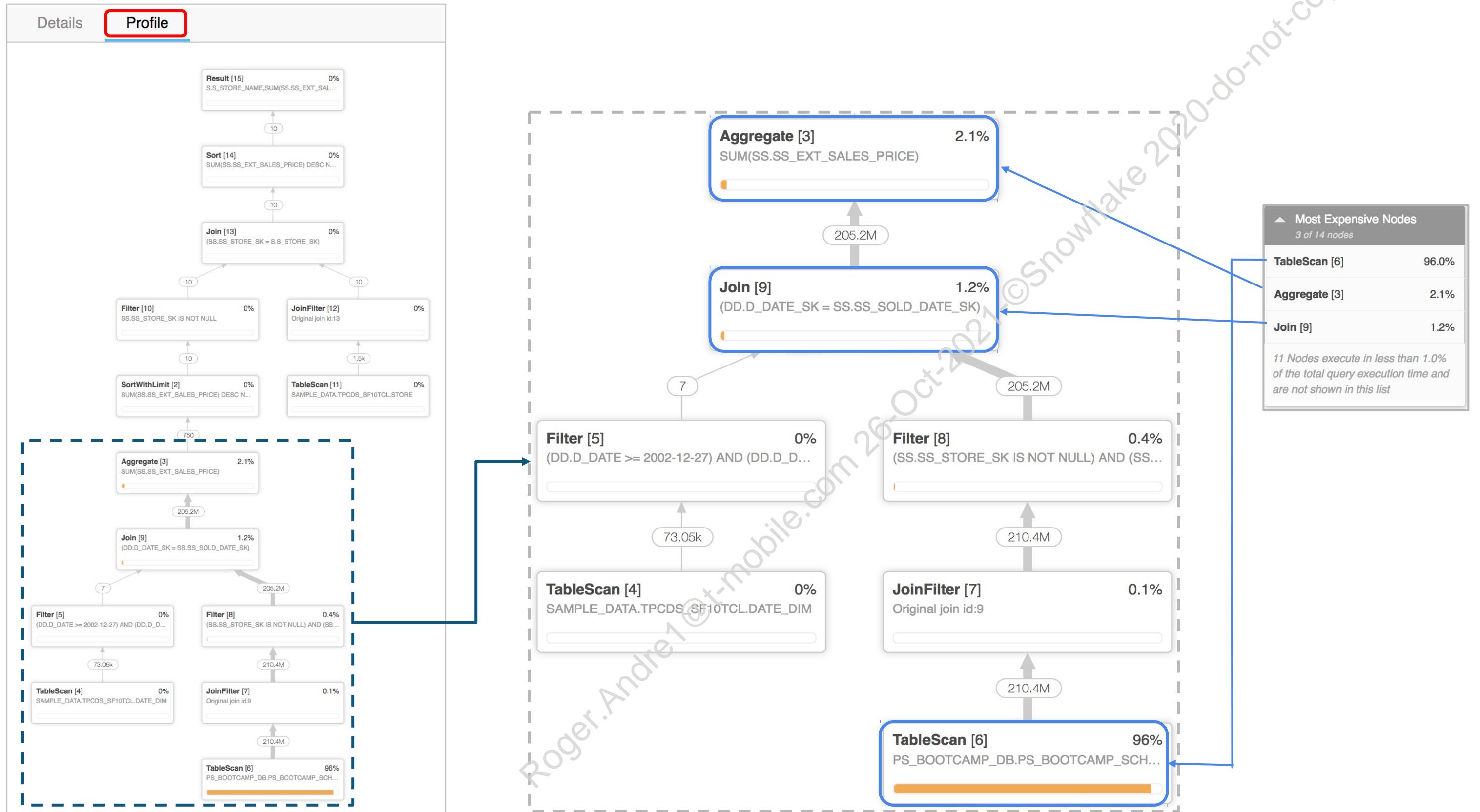
Each node in the query profile represents a step in the execution of the query



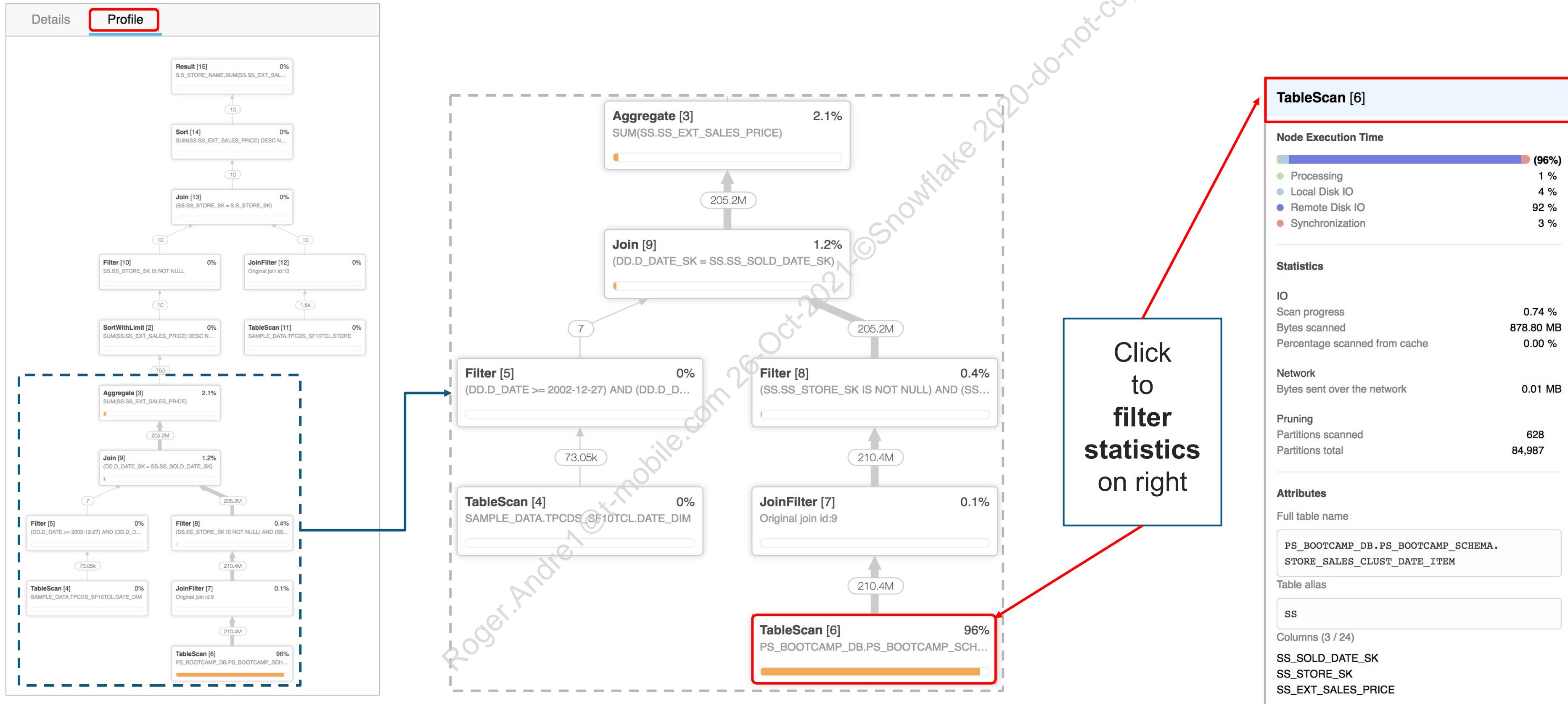
READING A QUERY PROFILE



READING A QUERY PROFILE

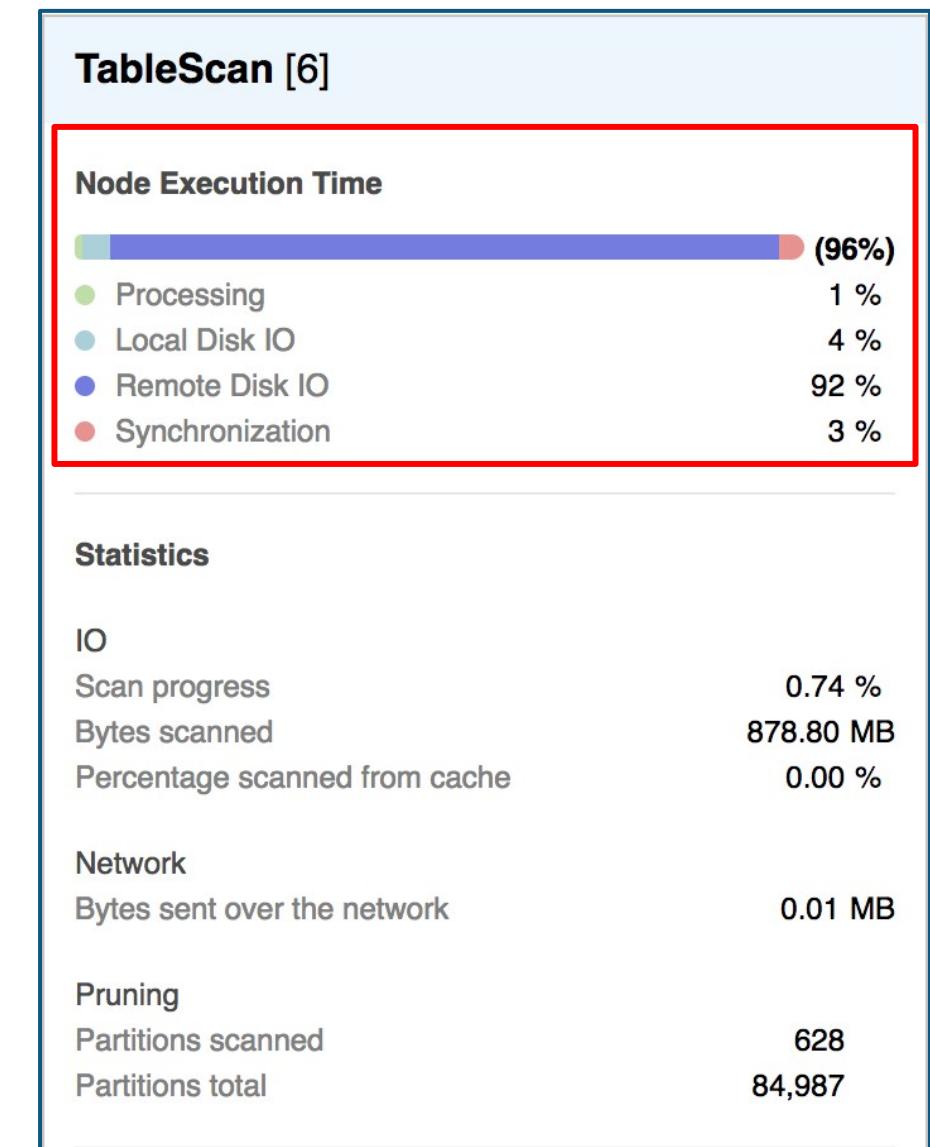


READING A QUERY PROFILE



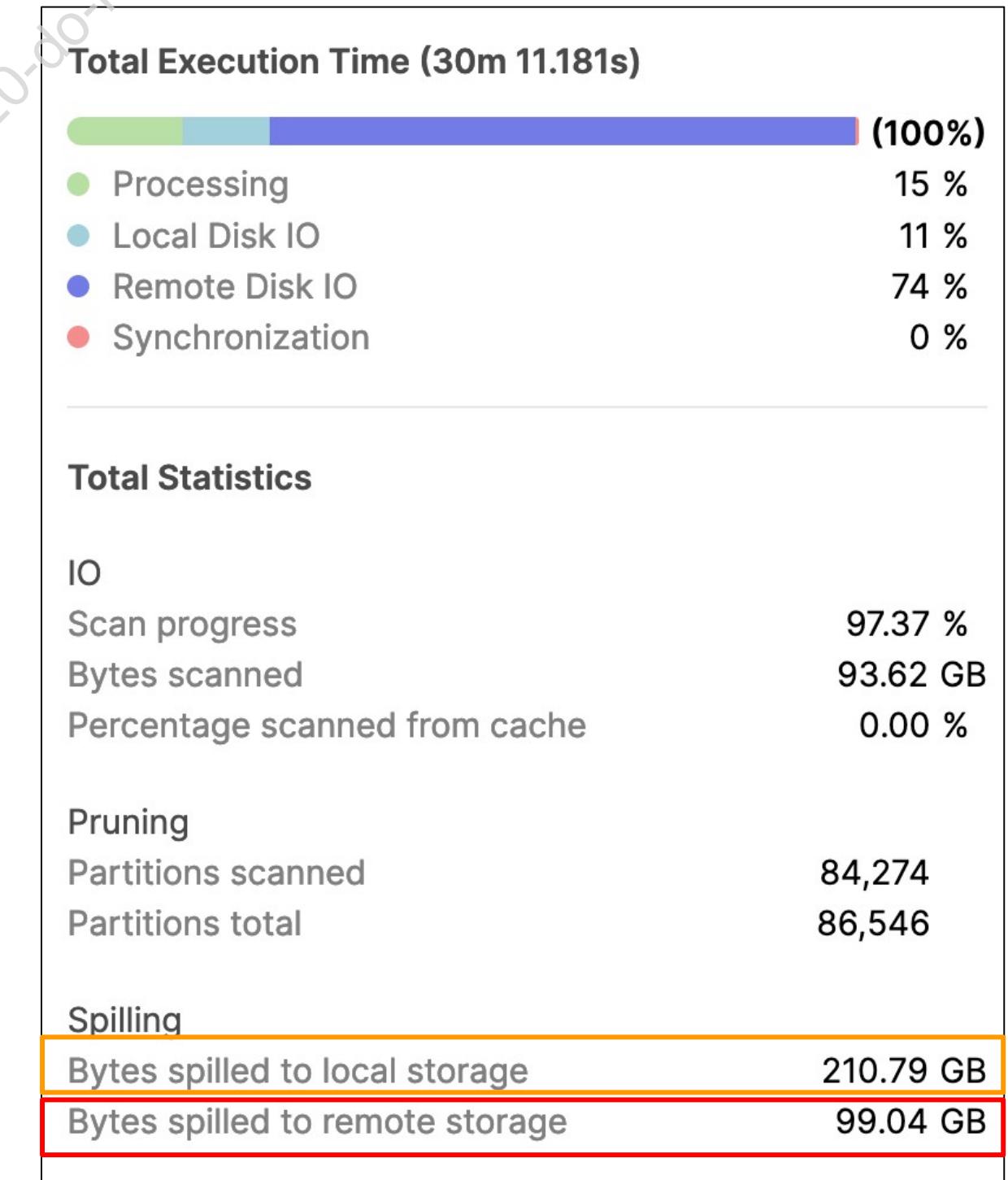
NODE EXECUTION TIME

- **Initialization** = setup activities prior to processing
- **Processing** = CPU data processing
- **Local Disk I/O** = blocked on local SSD on node
- **Remote Disk I/O** = blocked on remote cloud storage
- **Network Communication** = blocked on network data transfer
- **Synchronization** = sync activities between processes



STATISTICS

- **Scan progress** = % of data scanned for table thus far
- **Bytes scanned** = local + remote I/O
- **Percentage scanned from cache** = local / (local + remote)
- **External bytes scanned** = from external object (stage)
- **Bytes sent over network** = peer-peer data exchange
- **Partitions scanned** = number of micro-partitions read
- **Partitions total** = number of micro-partitions in table
- **Bytes spilled to local storage** = written to local SSD on node (insufficient memory)
- **Bytes spilled to remote storage** = written to remote cloud storage (insufficient local SSD)



LAB EXERCISE: 5

Review the Query Profile

15 minutes

Tasks:

- Run a query
- Review the query profile

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA LOADING AND UNLOADING

Roger.Andre1@t-mobile.com
26-Oct-2021 ©snowflake 2020-do-not-copy



MODULE AGENDA

- Data Loading
- Data Loading Recommendations
- Load Data Wizard
- Continuous Data Loading
- Data Unloading
- Data Transformation

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

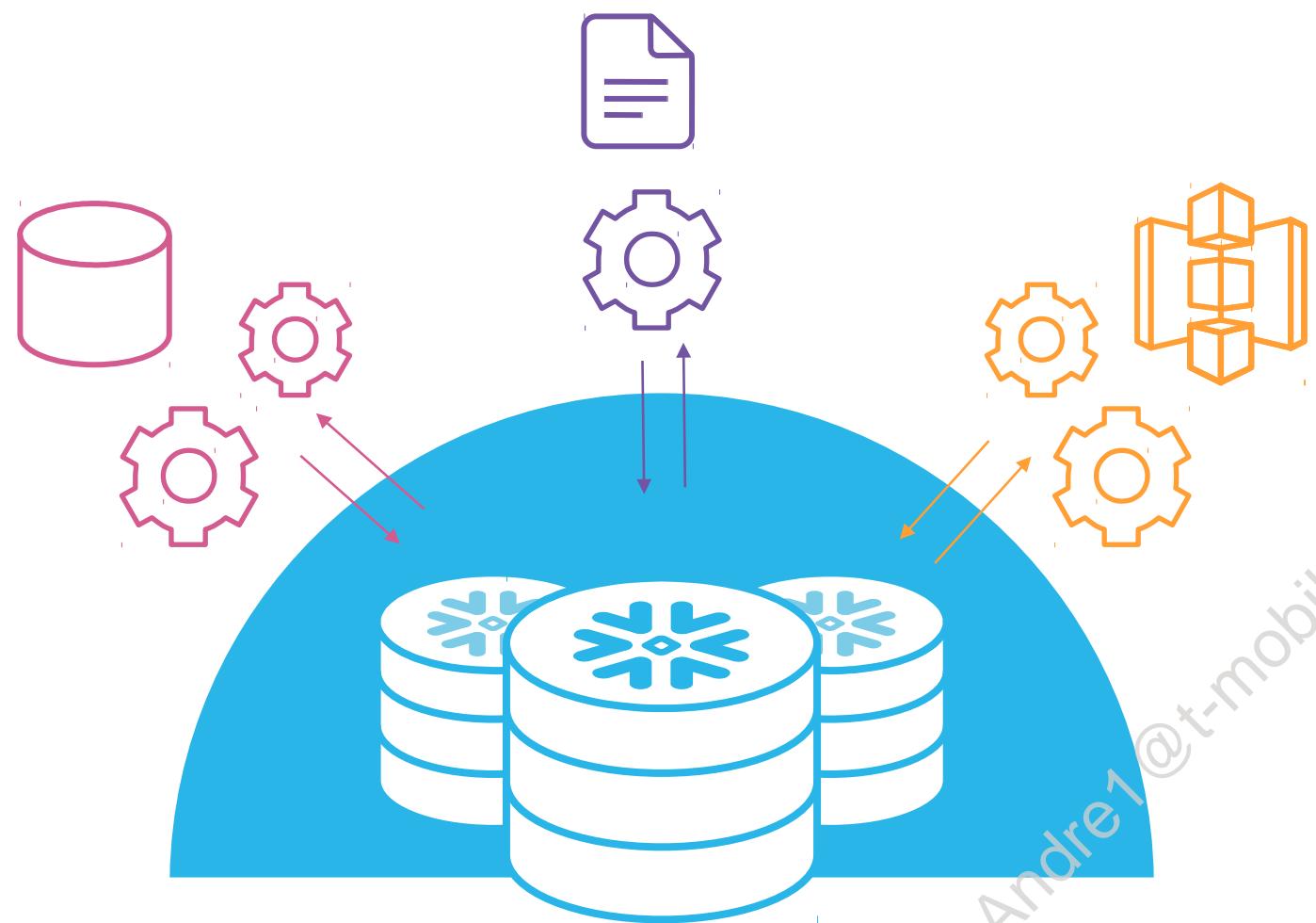


DATA LOADING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA LOADING



- High Level Data Loading Process
- What Happens During Load
- What is a Stage?
- What is a File Format?
- Data Loading Recommendations
- Syntax examples
- Load Data Wizard



HIGH LEVEL DATA LOADING PROCESS

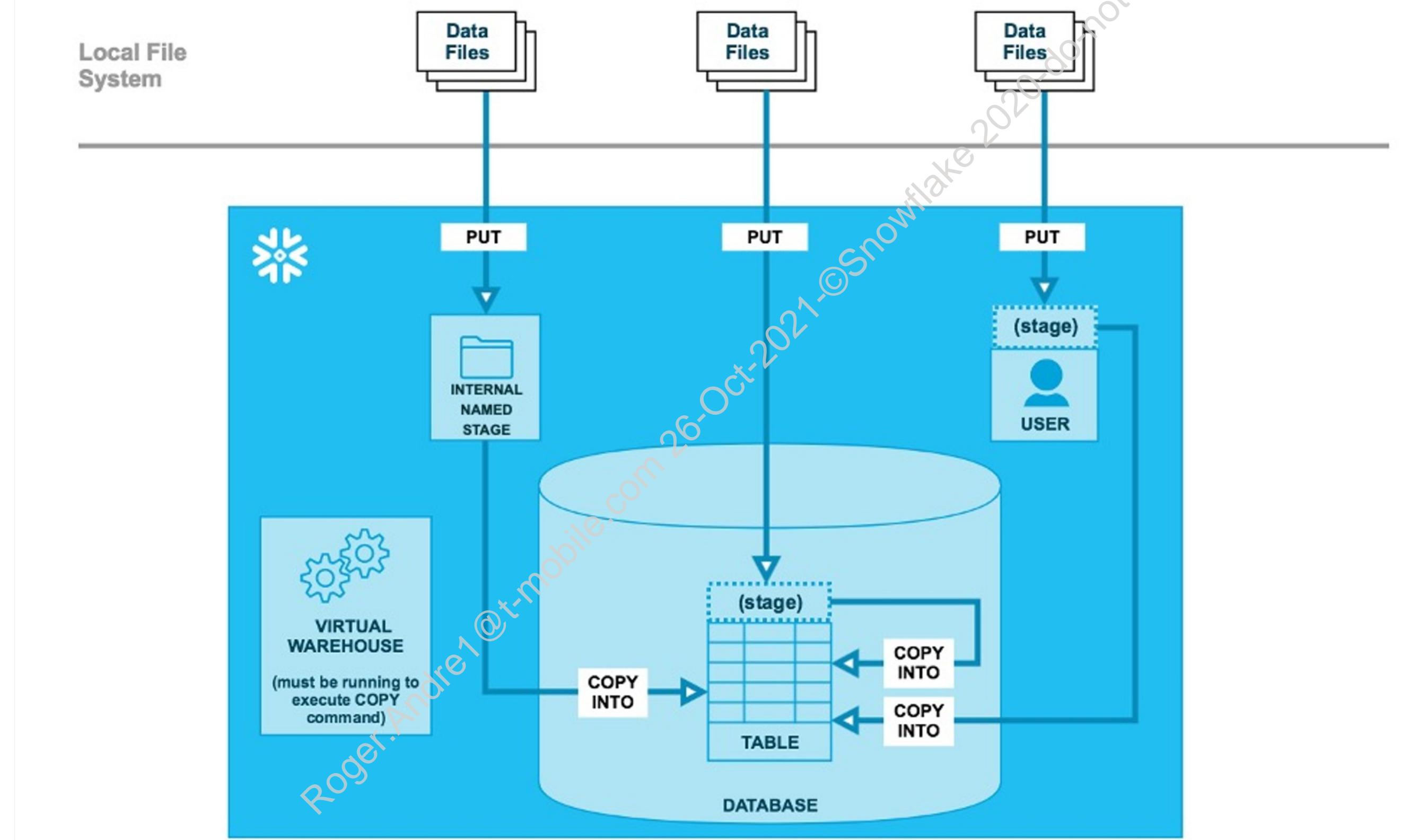


**Output Data from System
of Record to Files**

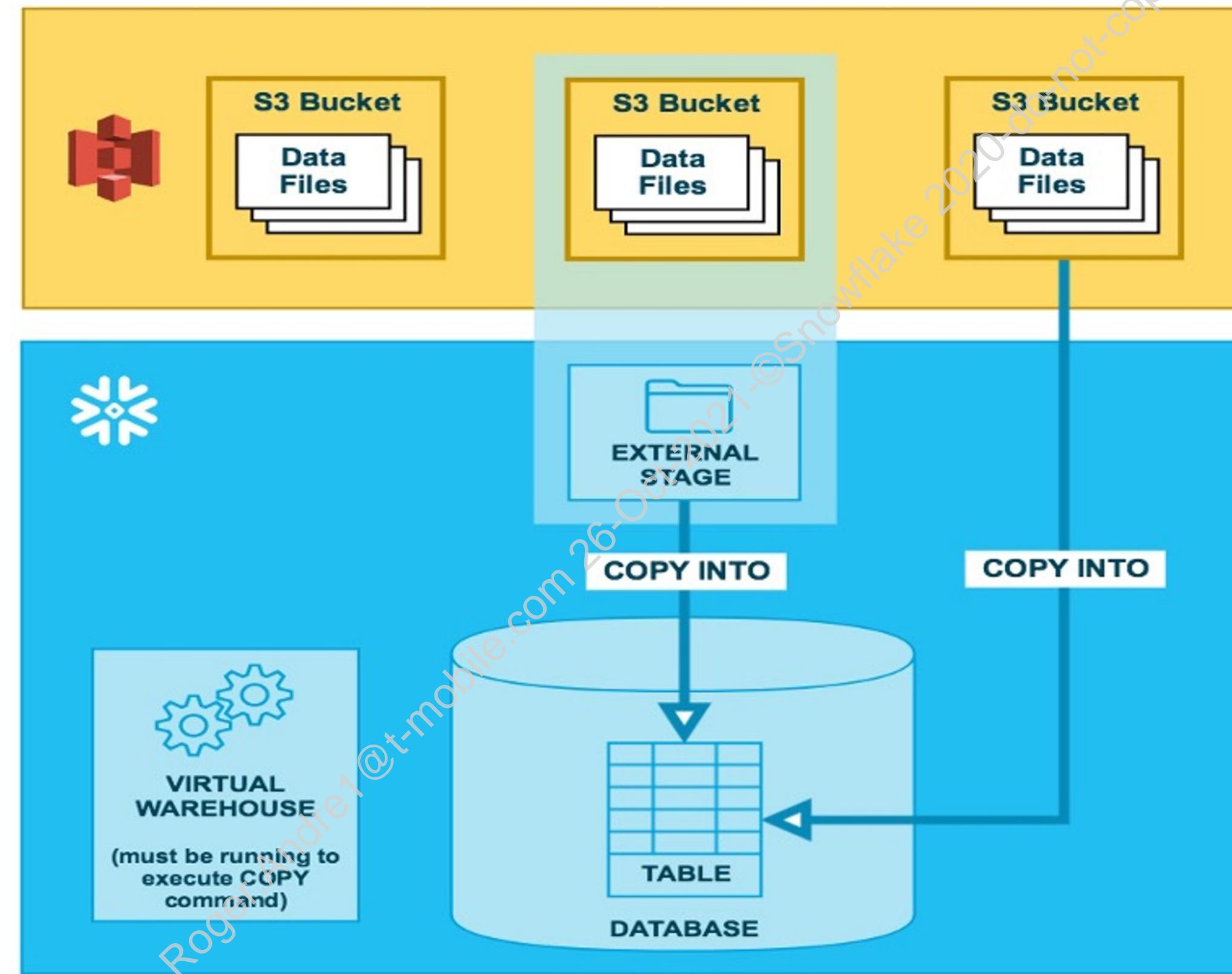
Stage Files to Cloud Storage

**Load Data from Cloud
Storage Into Snowflake**

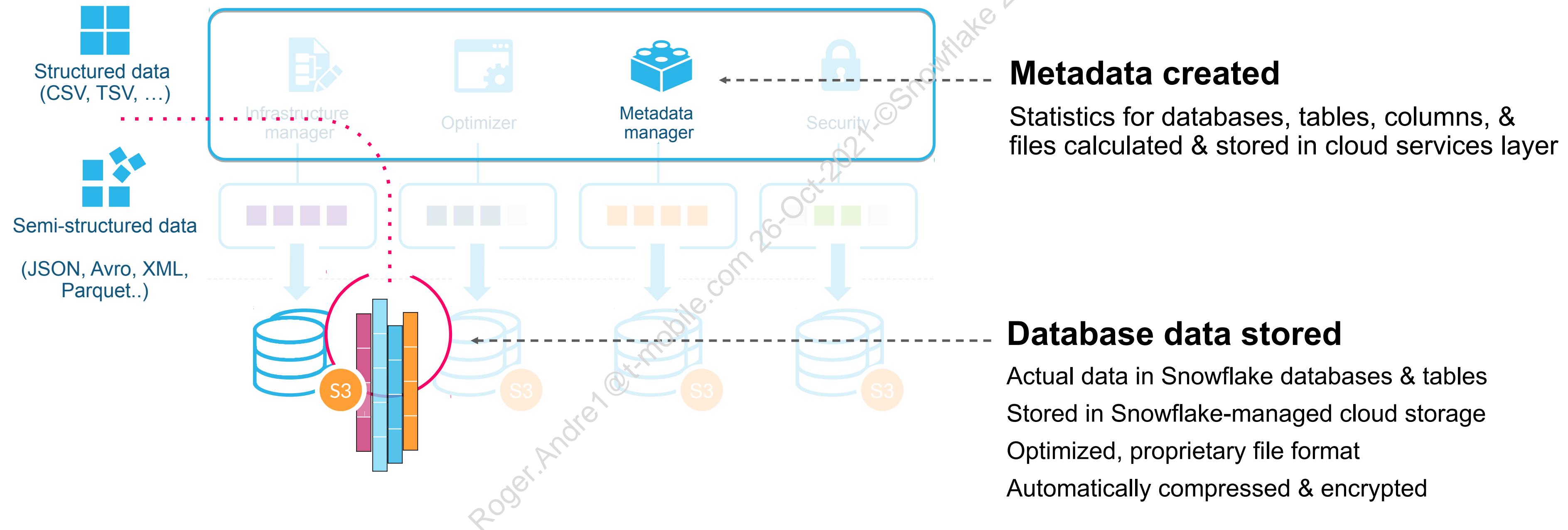
LOADING FROM LOCAL FILE SYSTEM



LOAD FROM CLOUD STORAGE



WHAT HAPPENS DURING LOAD



WHAT IS A STAGE?



Cloud file repository that simplifies and streamlines bulk loading and unloading

- Can be internal or external
 - Internal: stored internally in Snowflake
 - External: stored in an external location
- Best Practice: Create stage object to manage ingestion workload
 - Data can be queried directly from a stage without loading it first
- Snowflake supports AWS S3, Azure Blob, and Google as external locations



TYPES OF STAGES

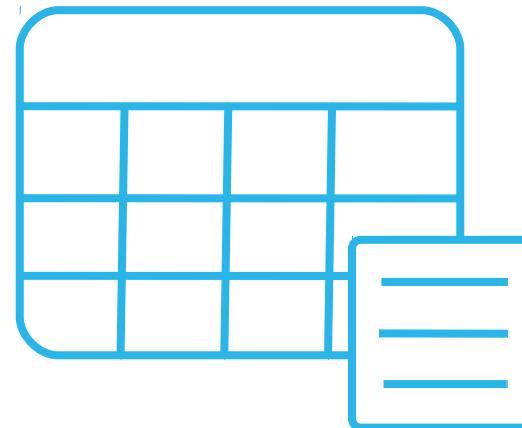
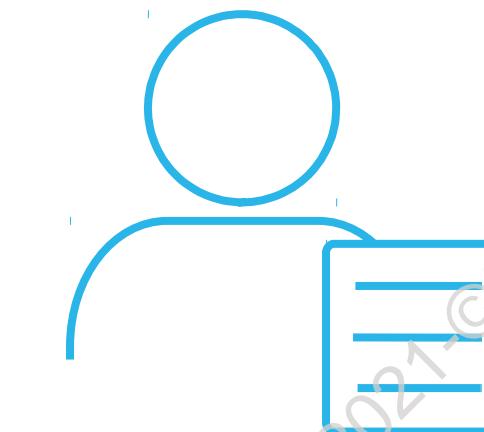
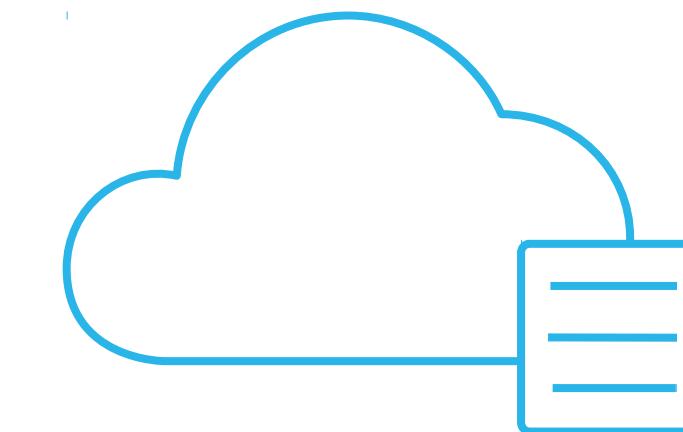


Table Stage
@%[TABLE_NAME]



User Stage
@~



Named Stage
@[STAGE_NAME]



- Created automatically
- Internal
- Can Specify file formats



- Created automatically
- Internal
- Does not support setting file formats



- Created manually
- Internal or External
- Can specify file format

WHAT IS A FILE FORMAT?

```
Col1,Col2,Col3,Col4  
123,ABC,987,FED  
234,BCD,876,EDC  
345,CDE,765,DCB
```

Named object that stores information needed to parse files during load/unload

- Specifies file type to load (CSV, JSON, etc.)
- Includes type-specific formatting options
- Specify FILE FORMAT object:
 - As part of a table
 - As part of a named stage
 - In the COPY INTO command
- File formats are used in this order:
 - Copy into, if available
 - Stage, if available
 - Table, if available
- File format not strictly required as some commands allow specifying the file format directly in the command.



SUPPORTED FILE FORMATS

- Structured
 - Delimited
 - CSV, TSV
- Semi-Structured
 - JSON – JavaScript Object Notation. Lightweight data-interchange format.
 - Avro – row-base storage format for Hadoop.
 - ORC – Optimized Row Columnar. Used to store Hive data.
 - Parquet – Columnar file format that stores binary data. Used in the Hadoop ecosystem
 - XML – Extensible Markup Language. Simple text-based format for representing structured information.

NOTE: Support for XML is currently in preview.



FILE FORMAT

```
CREATE FILE FORMAT <name>
```

```
  TYPE = CSV | JSON | AVRO | ORC | PARQUET | XML (Default if not specified is CSV)
```

```
  COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE | NONE
```

```
  FIELD_DELIMITER = '<character>' or NONE
```

```
  ERROR_ON_COLUMN_COUNT_MISMATCH = FALSE;
```



FILE FORMAT EXAMPLE

You'll be loading master data files from your ERP system into Snowflake. The source files are GZIP compressed csv files with a pipe delimiter. You don't want to generate an error if there is a mismatch between the columns in the file and the target table's columns.

```
CREATE FILE FORMAT master_data_gzip_pipe_erp
```

```
TYPE = ?
```

```
COMPRESSION = ?
```

```
FIELD_DELIMITER = ?
```

```
ERROR_ON_COLUMN_COUNT_MISMATCH = ? ;
```



FILE FORMAT EXAMPLE

You'll be loading master data files from your ERP system into Snowflake. The source files are GZIP compressed csv files with a pipe delimiter. You don't want to generate an error if there is a mismatch between the columns in the file and the target table's columns.

```
CREATE FILE FORMAT master_data_gzip_pipe_erp
  TYPE = CSV
  COMPRESSION = GZIP
  FIELD_DELIMITER = '|'
  ERROR_ON_COLUMN_COUNT_MISMATCH = FALSE;
```



COPY INTO

- Requires an active warehouse to execute
- Can specify copy options to control how errors are handled:

copyOptions ::=

```
ON_ERROR = { CONTINUE | SKIP_FILE | SKIP_FILE_<num> |  
            SKIP_FILE_<num>% | ABORT_STATEMENT }
```

- Regular expressions can be used to filter the files to be loaded



COPY INTO

```
COPY INTO <table_name>
```

```
FROM @database.schema.stage/[file path]/
```

(refers to Internal Stage | External Stage | External Location)

```
FILES = ('<file_name>')
```

```
FILE_FORMAT = (FORMAT_NAME = <file_format_name>)
```



COPY INTO EXAMPLE

You want to copy data from a file (*marketing_data.tbl*) in an internal named stage into a table called *marketing*. Assume you've set your context to the database and schema shown below.

Database: *db_international*

Schema: *schema_marketing*

Internal Named Stage: *data_import*

File path: */load/marketing/2021/01/01*

File format: *marketing_gzip_pipe*

```
COPY INTO ?
```

```
FROM ?
```

```
FILES = ('?')
```

```
FILE_FORMAT = (FORMAT_NAME = ?)
```



COPY INTO EXAMPLE

You want to copy data from a file (*marketing_data.tbl*) in an internal named stage into a table called *marketing*. Assume you've set your context to the database and schema shown below.

Database: *db_international*

Schema: *schema_marketing*

Internal Named Stage: *data_import*

File path: */load/marketing/2021/01/01*

File format: *marketing_gzip_pipe*

```
COPY INTO marketing
```

```
FROM @data_import/load/marketing/2021/01/01
```

```
FILES = ('marketing_data.tbl')
```

```
FILE_FORMAT = (FORMAT_NAME = marketing_gzip_pipe)
```



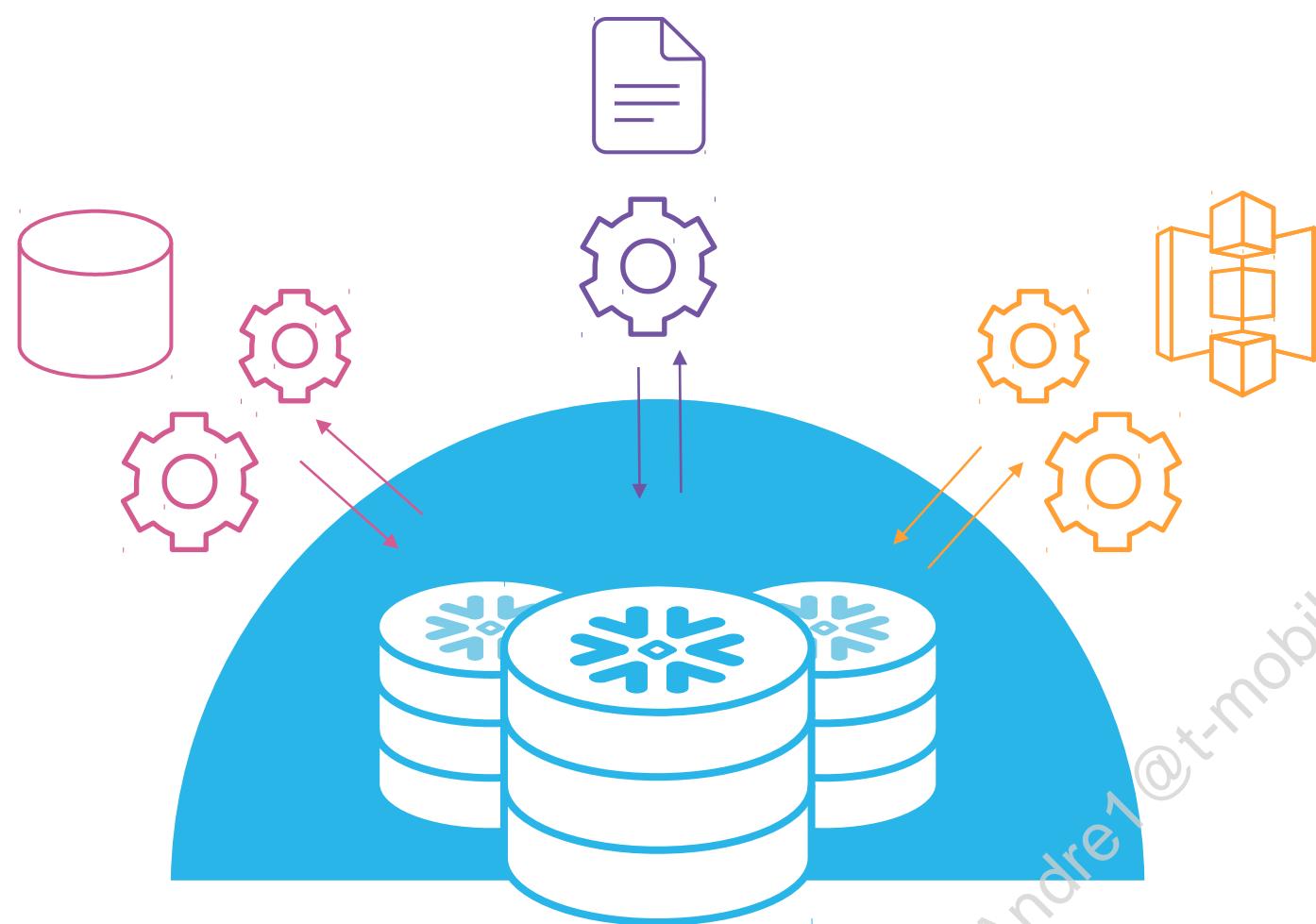
DATA LOADING RECOMMENDATIONS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



RECOMMENDATIONS

DATA LOADING



- File size
- Location path



FILE SIZE AND NUMBER

- File size and number of files are crucial to optimizing load performance
- Split large files before loading into Snowflake
- Recommended: 100MB to 250MB (compressed)

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 Do-not-copy



FILE SIZE AND NUMBER

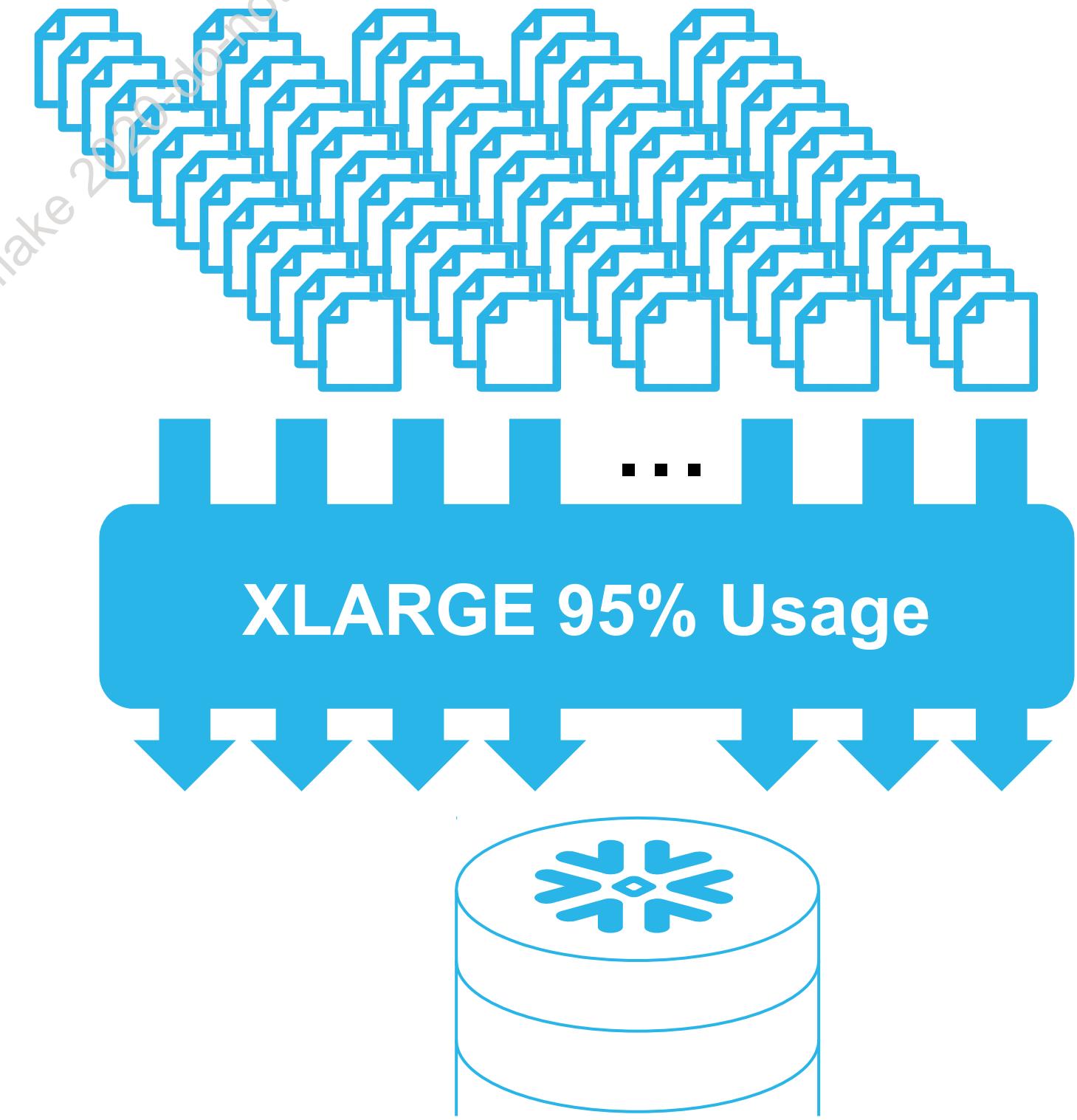
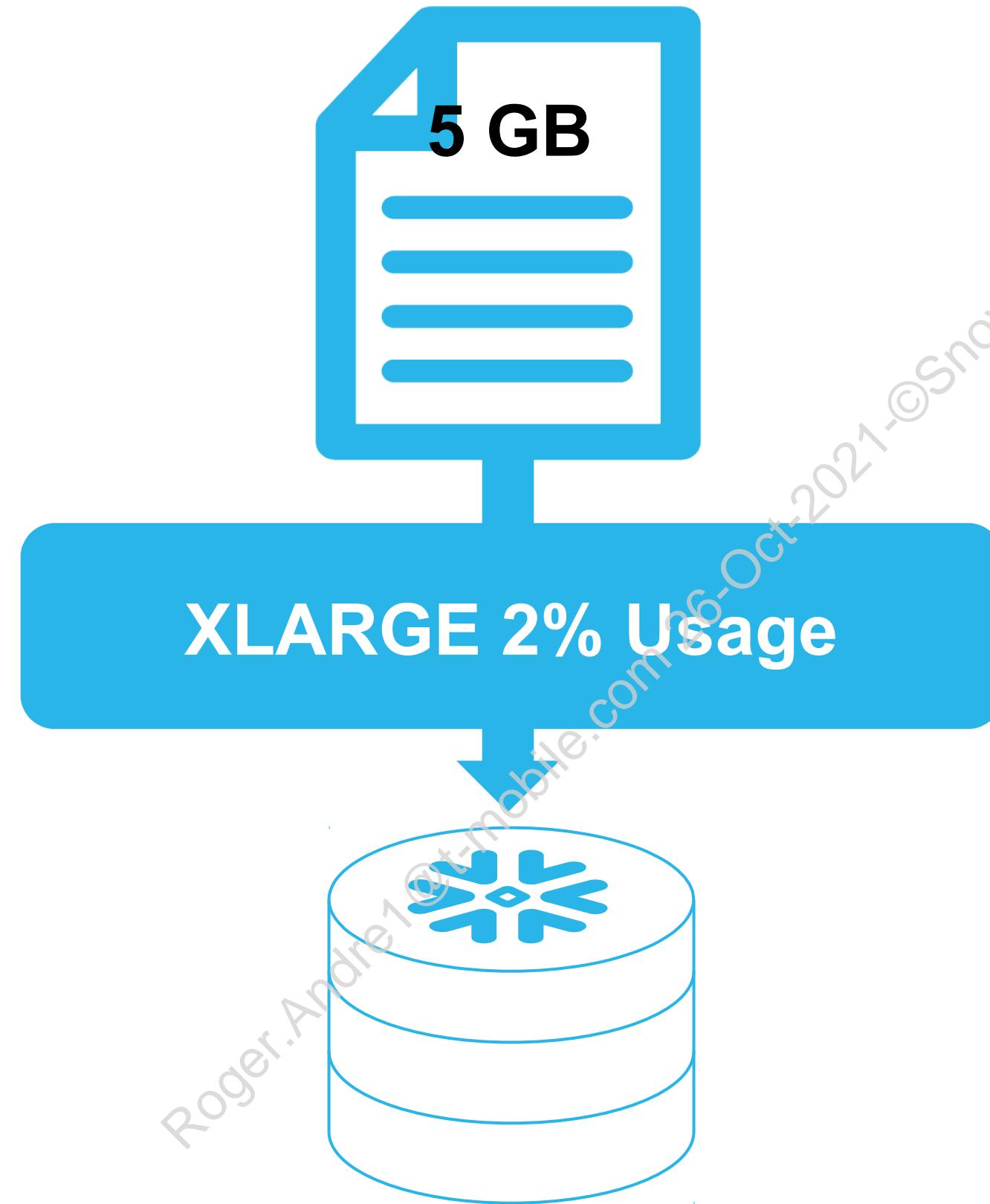
- File size and number of files are crucial to optimizing load performance
- Split large files before loading into Snowflake
- Recommended: 100MB to 250MB (compressed)

Warehouse Size	# files in parallel
XS	8
S	16
M	32
L	64
XL	128



SERIAL COPY VS PARALLEL COPY

Data Files
Virtual Warehouse
Snowflake Tables



Note: Single COPY Command



FILE ORGANIZATION

- Organize data in logical paths (e.g., subject area and create date)

/system/market/daily/2018/09/05/

- Use wildcards late in the file path definition, to reduce scanning:

```
COPY INTO table FROM /system/market/daily/2018/*
```



FILE LOCATIONS

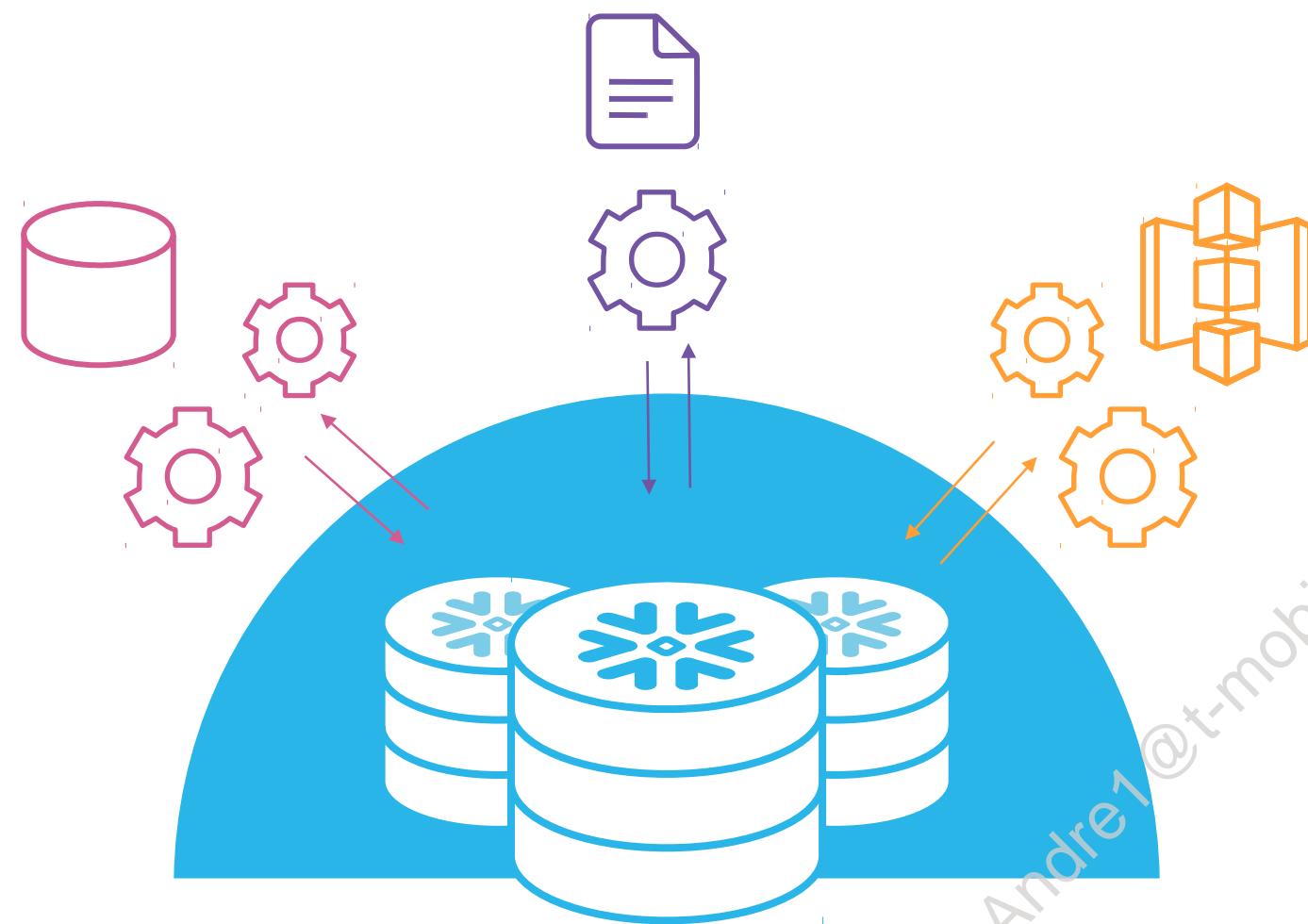


Single Location with Many Files
Slower: must scan more files to find needed files



Many Locations with Fewer Files
Faster: targeted directories allow for scanning fewer files

SYNTAX



- File Format Syntax
- COPY INTO Syntax
- Load Data Wizard



LOAD DATA WIZARD

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



LOAD DATA WIZARD

- Offers a graphic user interface for loading limited amounts of data
- Deletes all staged files after the load completes
- Can load from a stage or from your computer
- Can load structured or semi-structured data
- Wizard is only for loading small numbers of files of up to 50 MB



LOAD DATA WIZARD

The screenshot shows the Snowflake web interface. At the top, there's a navigation bar with icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets, and History. The 'Databases' icon is highlighted with a red circle and labeled '1.'. Below the navigation bar, the path 'Databases > CAMEL_DB' is displayed. Underneath, there are tabs for Tables, Views, Schemas, Stages, File Formats, Sequences, and Pipe. The 'Tables' tab is selected and highlighted with a blue underline. In the main area, there's a table with columns: Table Name, Schema, Creation Time, and Owner. Two rows are visible: 'EMPLOYEES' (selected with a red circle and labeled '2.') and 'NATION'. At the bottom of the table area, there are buttons for Create..., Create Like..., Clone..., Load Data..., Drop..., and Transfer. The 'Load Data...' button is highlighted with a red circle and labeled '3.'.

Table Name	Schema	Creation Time	Owner
EMPLOYEES	PUBLIC	1/12/2021, 2:58:05 ...	TRAINING_ROLE
NATION	CAMEL_LAB	1/12/2021, 9:38:07 ...	TRAINING_ROLE

Scenario: Loading GZIP, pipe-delimited data from a file on your computer. The file contains employee records from your ERP system.

- Click Databases
- Select the table row
- Click Load Data to launch the wizard

WAREHOUSE

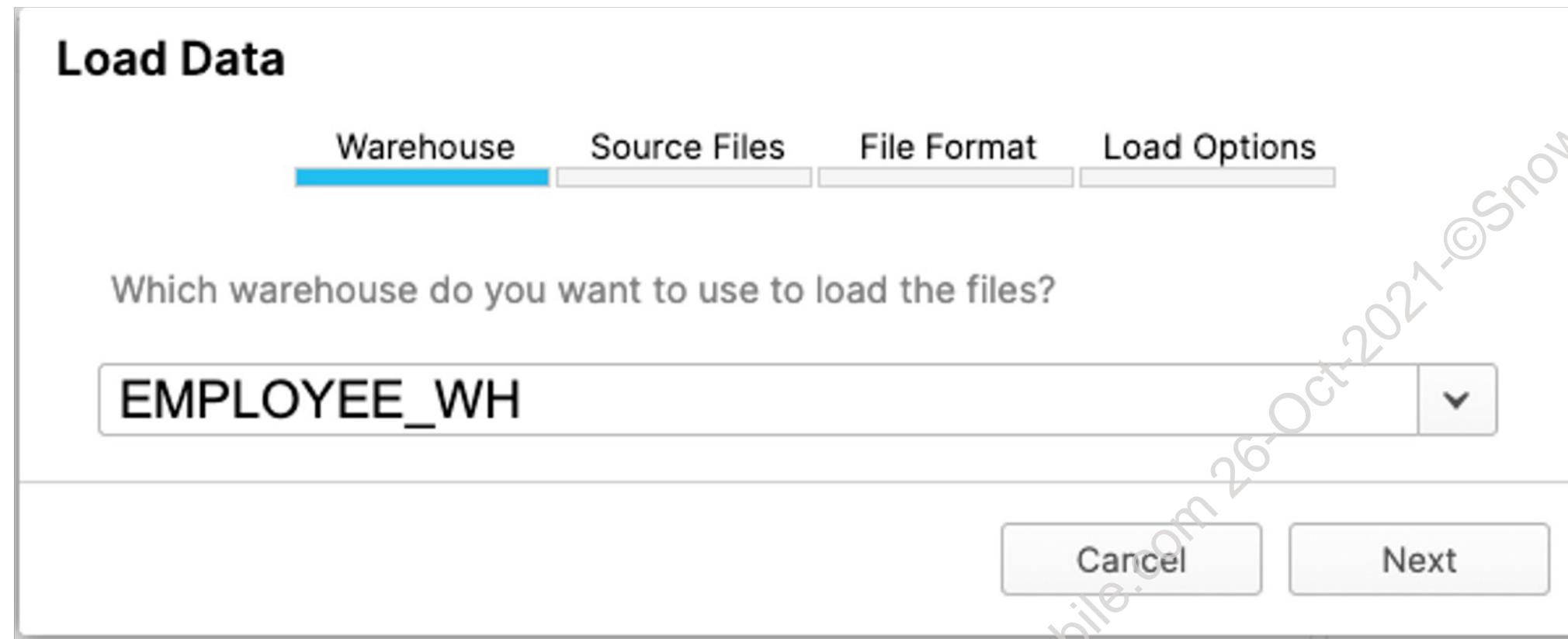
Load Data

Warehouse Source Files File Format Load Options

Which warehouse do you want to use to load the files?

EMPLOYEE_WH

Cancel Next



- Select a warehouse
- Click Next



SOURCE FILES

Load Data

Warehouse **Source Files** File Format Load Options

From where do you want to load files?

Load files from your computer
Select Files...

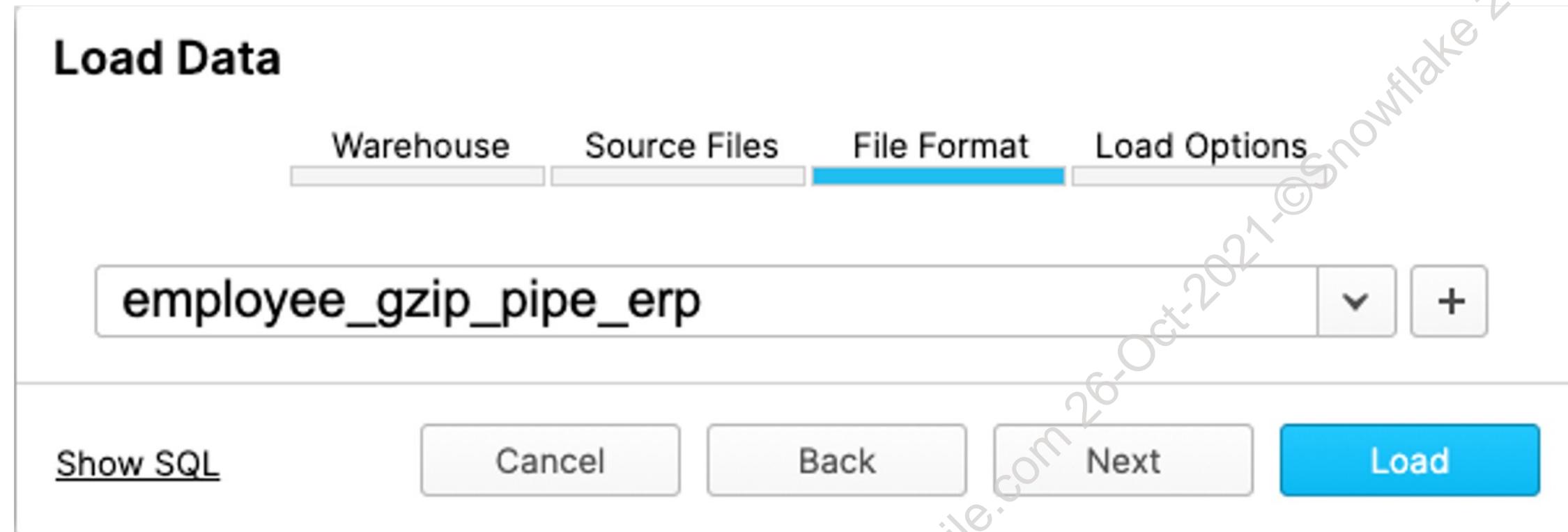
Load files from external stage
Stage: [dropdown] +
Path: [input field]

Cancel Back Next

- Click the Select Files button to navigate to and select your file
- Click Next



FILE FORMAT



- Select an available file format
- Click Next

LOAD OPTIONS

Load Data

Warehouse Source Files File Format **Load Options**

What should the load do if it encounters an error while parsing a file?

Do not load any data in the file
 Stop loading, rollback and return the error
 Do not load any data in the file if the error count exceeds:
Threshold ?

Continue loading valid data from the file

[Show SQL](#) Cancel Back Load

- Select a load option
- Click Load

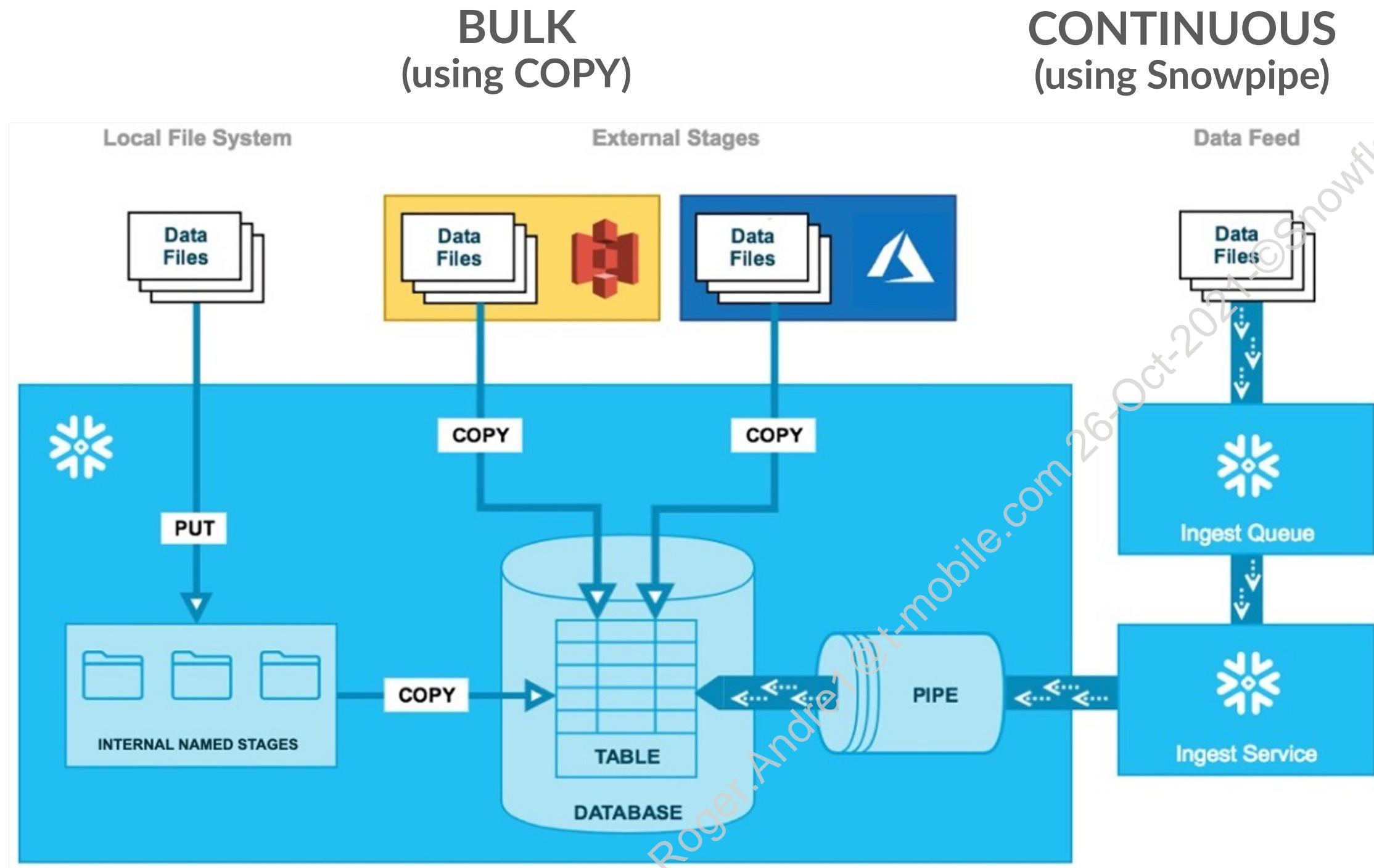


CONTINUOUS DATA LOADING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy



DATA LOADING APPROACHES



- Snowpipe can reference an internal/external stage or local file system
- Pipes are created using CREATE OR REPLACE. This resets the pipes load history to empty.

USE CASE

COPY command (BATCH)

- Migration from traditional data sources
- Transaction boundary control
 - BEGIN / START TRANSACTION / COMMIT / ROLLBACK
- Independently scale compute resources for different ingestion workloads

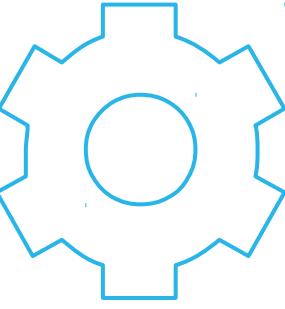
Snowpipe (CONTINUOUS)

- Ingestion from modern data sources
- Continuously generated data is available for analysis in seconds
- No scheduling (with auto-ingest)
- Serverless model needs no user-managed virtual warehouse



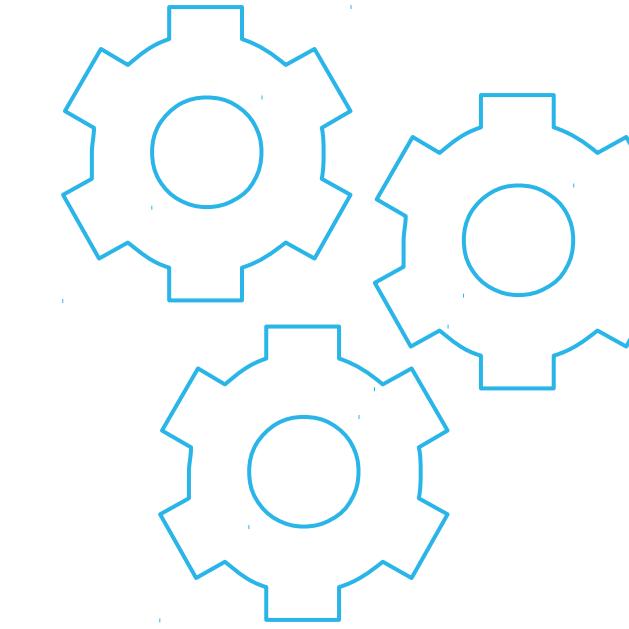
REST VS AUTO_INGEST

REST



- Manually calls the REST API endpoint
- Passes a list of files in the stage
- Works with internal or external stages

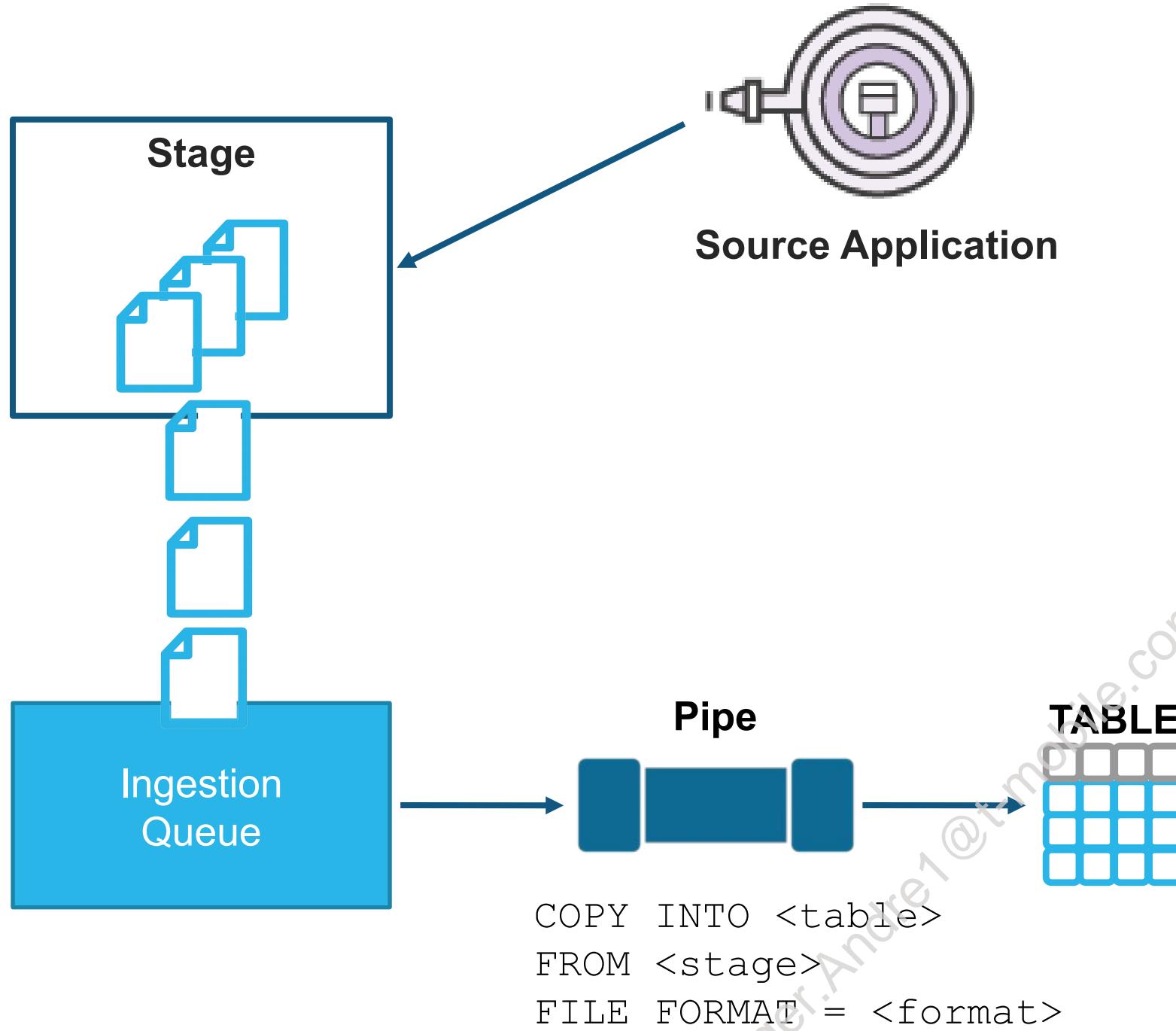
AUTO_INGEST



- Receive notifications for new files
- “Wakes up” and processes new files
- Works only with external stages



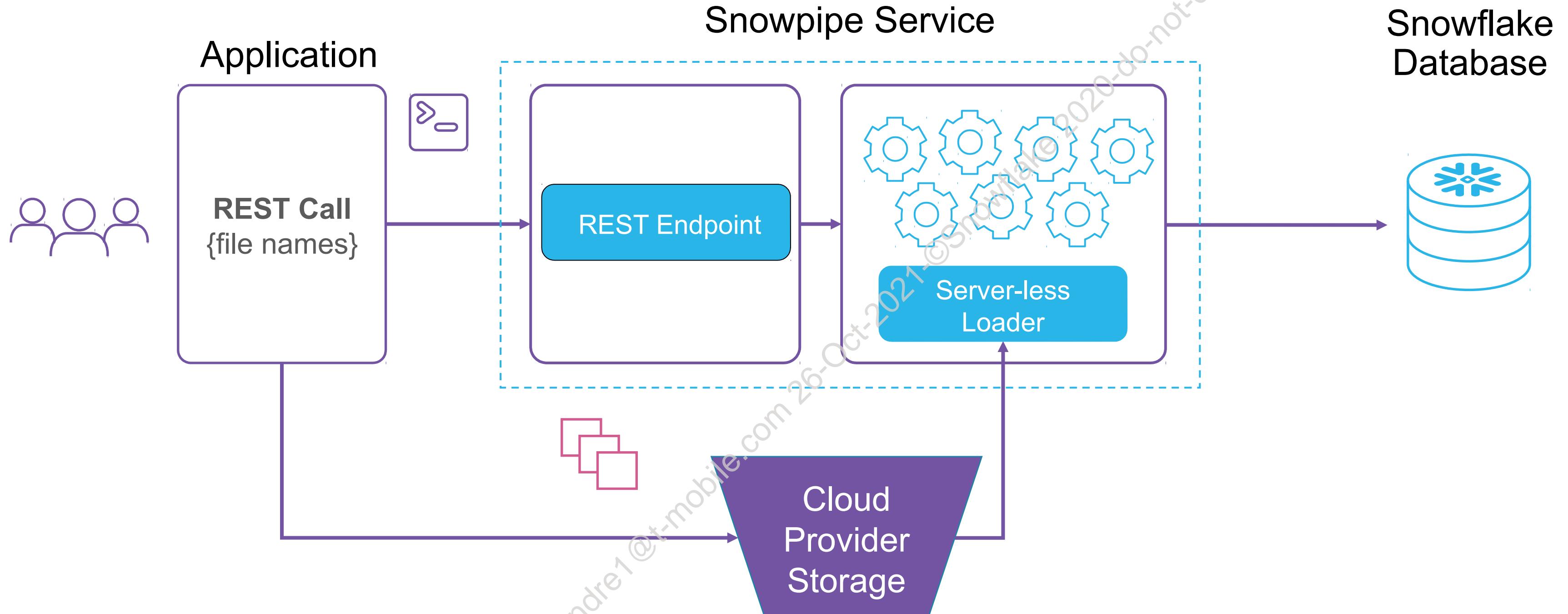
PIPE



- Source application loads stage
- Files moved from stage to ingestion queue
- Pipe contains a **COPY INTO** statement
 - Source stage for data files
 - Target table
- Loads data into tables continuously from an ingestion queue
- Can be suspended/resumed, return status
- When pipes are created using CREATE OR REPLACE, their load history is reset to empty



SNOWPIPE REST API



```
CREATE PIPE IF NOT EXISTS mypipe AS COPY INTO mytable FROM @mystage;
```

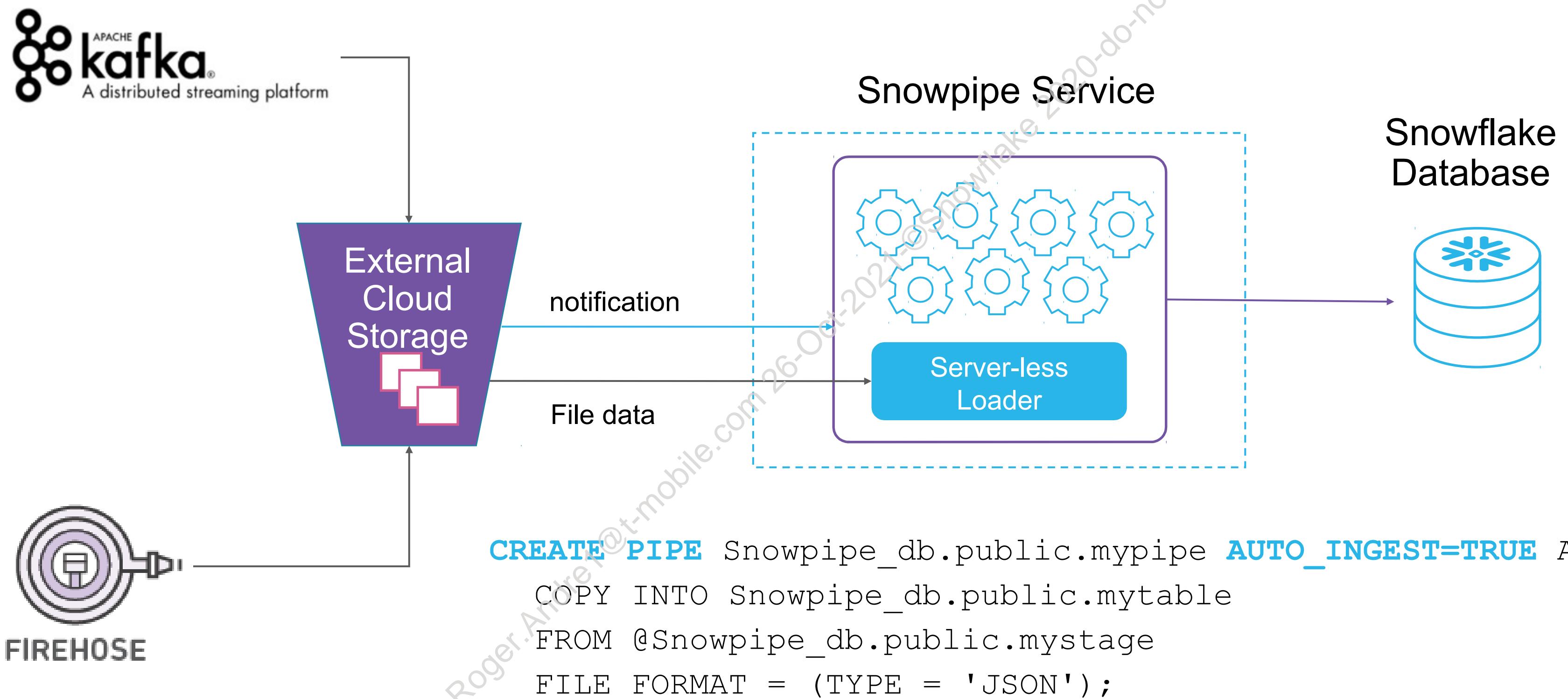


SNOWPIPE REST API TIPS

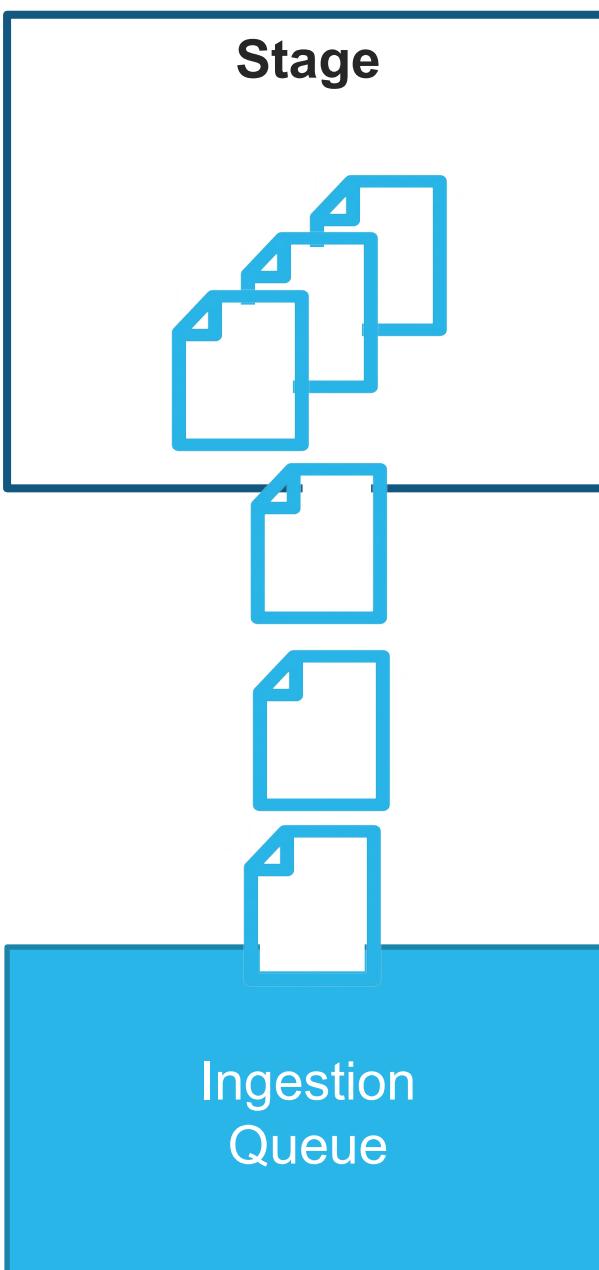
- Snowflake manages the compute required to execute COPY INTO commands
- Snowpipe keeps track of which files it has loaded
- When pipe is recreated using CREATE OR REPLACE PIPE, the load history is reset to empty



AUTO-INGEST



FILE LOAD ORDER



- Staged files are moved into an ingest queue
- Files that appear in the stage later are appended to the queue
- Multiple processes pull from the queue
- Older files are generally loaded first, but files are not guaranteed to be loaded in the same order they are staged



SNOWPIPE RECOMMENDATIONS

- File sizes 100-250MB compressed
- Stage files no more than once per minute
 - Overhead to manage files increases in relation to the number of files queued
- Currently supported on Amazon AWS, Microsoft Azure and Google Cloud Platform



SNOWPIPE BILLING

- Serverless model: does not require a virtual warehouse
- Snowflake provides and manages compute resources
 - Capacity grows and shrinks depending on load
- Accounts charged based on actual compute usage
 - Don't need to worry about suspending a warehouse
 - Charged per-second, per-core
- Utilization cost of 0.06 credits per 1000 files notified via REST calls or auto-ingest

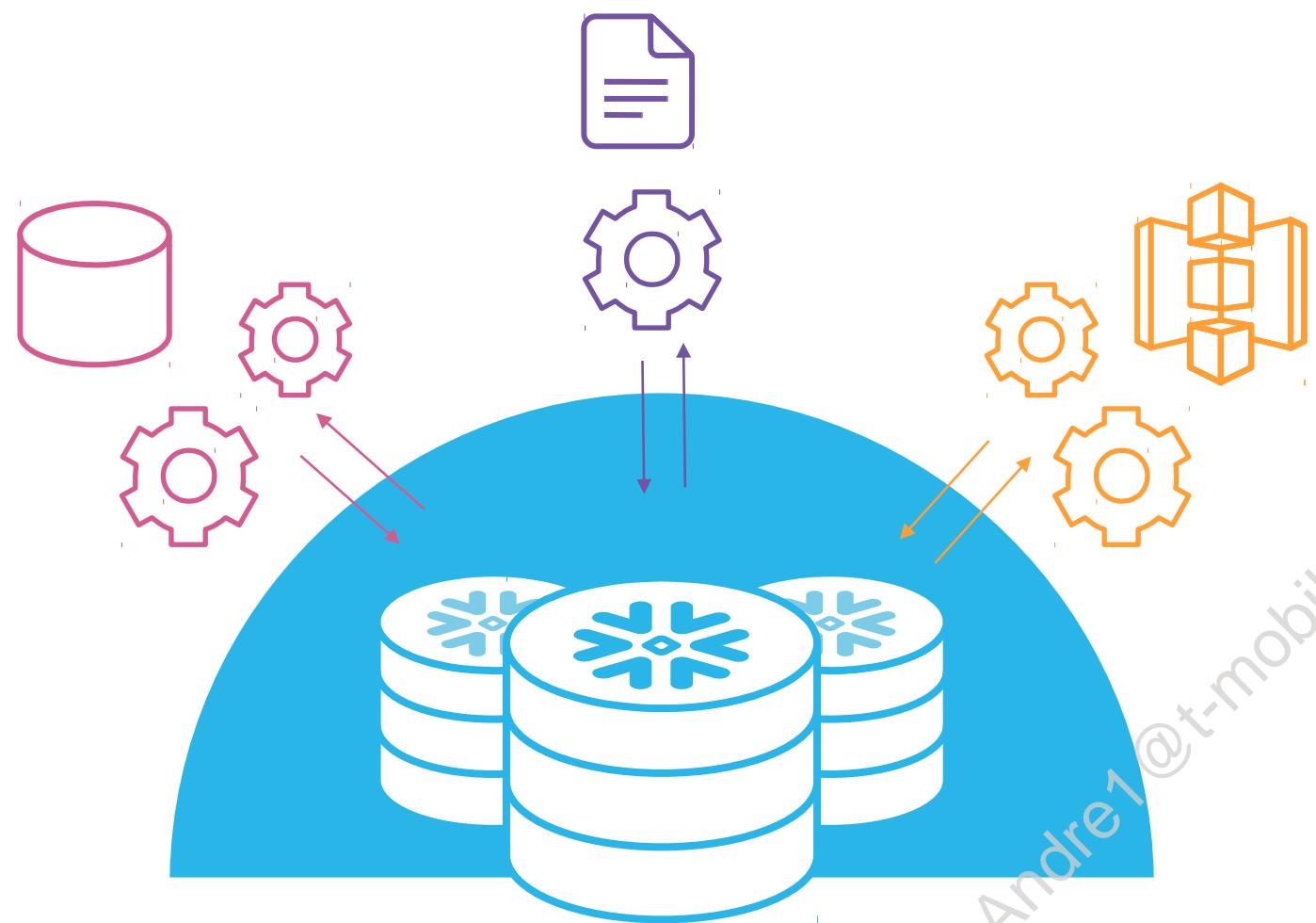


DATA UNLOADING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA UNLOADING

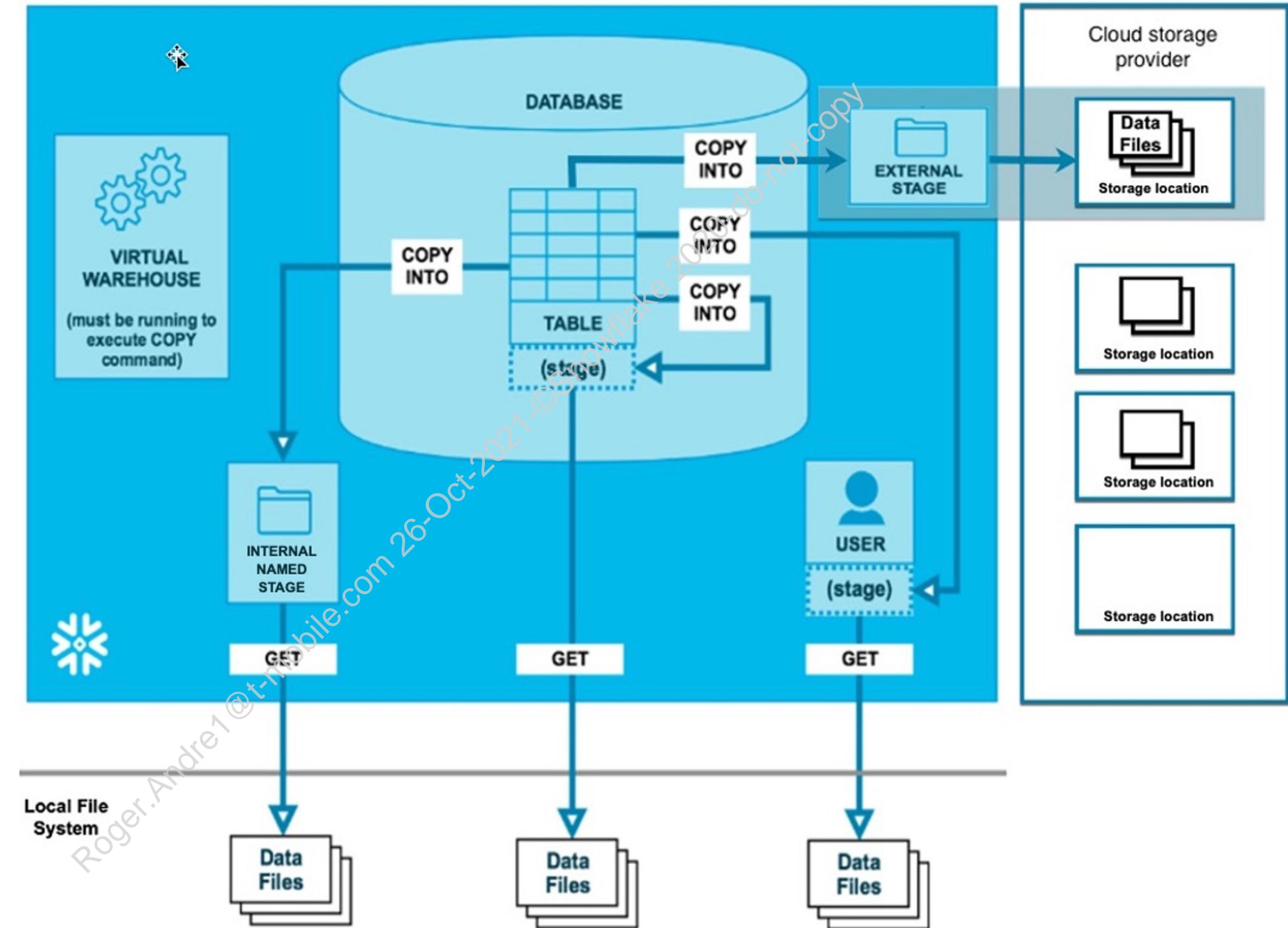


- Unload Destinations
- Unload with SELECT
- File Paths and Names
- LIST Command
- REMOVE Command

UNLOAD DESTINATIONS

Unload to:

1. Table Stage
 2. Internal named stage
 3. External stage



UNLOAD SYNTAX

- Syntax for internal or external stage unload:

```
COPY INTO @my_stage  
FROM my_data  
FILE_FORMAT = (FORMAT_NAME = 'my_format');
```

- Can use any SQL command and express joins:

```
COPY INTO @my_stage  
FROM (SELECT column1, column2  
      FROM my_table m JOIN your_table y ON m.id = y.id)  
FILE_FORMAT = (FORMAT_NAME = 'my_format');
```

- Use @% to unload into a table stage:

```
COPY INTO @%my_table  
FROM (SELECT column1, column2 FROM my_table)  
FILE_FORMAT = (FORMAT_NAME = 'my_format');
```



FILE PATHS AND NAMES

- To set the file path for the files, add the path after the stage name.

```
COPY INTO @mystage/CSVFiles/ FROM mytable
```

- To set the file name for the files, add the name after the folder name.

```
COPY INTO @mystage/CSVFiles/testfile FROM mytable
```

- Snowflake appends a suffix to the file name that includes:
 - The number of the virtual machine in the virtual warehouse
 - The unload thread
 - A file number.

Example: **testfile_0_0_0.csv**



UNLOAD TIPS

- Can unload into any flat, delimited plain text format (CSV, TSV, etc.)
- JSON data must be unloaded from a column of VARIANT data type
- A SELECT statement can be used to unload a table to multi-column, semi-structured format (Parquet or JSON only)
- Can also unload the file in compressed format

AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE | NONE



LIST COMMAND

- Use the LIST command to list the files stored in a stage object:

```
LIST @%monthly_sales_agg;
```

name	size	md5	last_modified
data	144	055c5ee1d38271580a6a925ad1a69d45	Sun, 2 Dec 2018 17:48:22 GMT
data_0_0_0.csv	320	aefac11b7e3b543809988ea3f90924c5	Sun, 2 Dec 2018 18:15:06 GMT
data_0_0_0.csv.gz	144	0f01461eab63268987260ae6622c3ec6	Sun, 2 Dec 2018 18:15:19 GMT
data_0_0_0.json.gz	208	72e0f0d577b590233cda48464184204b	Sat, 1 Dec 2018 16:44:43 GMT
monthly_sales_dec_0_0_0.json.gz	208	37888eaf7698a6ac9eee178eaa509e57	Sat, 1 Dec 2018 16:47:26 GMT



REMOVE COMMAND

- Use the REMOVE command to delete files stored in an internal stage:

```
REMOVE @%monthly_sales_agg PATTERN='.*data.*';
```

name	result
data_0_0_0.json.gz	removed
data_0_0_0.csv.gz	removed
data	removed
data_0_0_0.csv	removed

```
LIST @%monthly_sales_agg;
```

name	size	md5	last_modified
monthly_sales_dec_0_0_0.json.gz	208	37888eaf7698a6ac9eee178eaa509e57	Sat, 1 Dec 2018 16:47:26 GMT



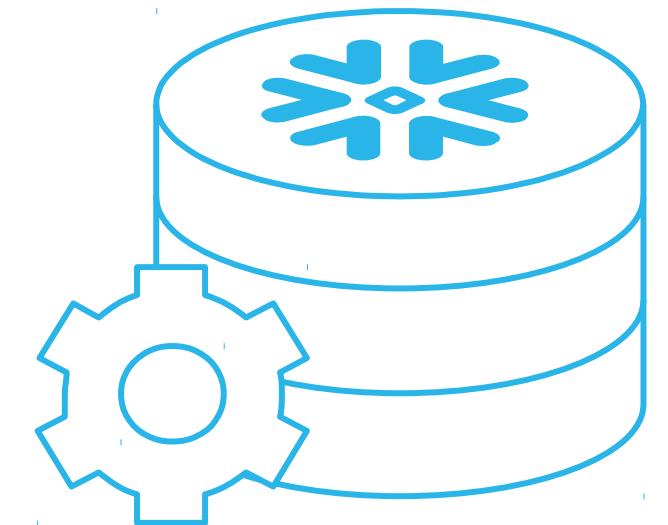
DATA LOADING TRANSFORMATIONS AND MONITORING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



TRANSFORMING DATA DURING LOAD

- The COPY command supports column reordering, column omission, and CAST using a SELECT statement
 - NOT SUPPORTED: Joins, filters, aggregations
 - Can include SEQUENCE columns, current_timestamp(), or other column functions during data load
- The VALIDATION_MODE parameter does not support transformations in COPY statements



TRANSFORMING DATA DURING LOAD EXAMPLES

```
COPY INTO home_sales (city, zip, sale_date, price)
FROM (SELECT SUBSTR(t.$2,4), t.$1, t.$5, t.$4 FROM @my_stage t)
FILE_FORMAT = (FORMAT_NAME = MYCSVFORMAT);
```

```
COPY INTO casttb(col1, col2, col3)
FROM (SELECT TO_BINARY(t.$1, 'utf-8'),
TO_DECIMAL (t.$2, '99.9', 9, 5), TO_TIMESTAMP_NTZ(t.$3)
FROM @~/datafile.csv.gz t) FILE_FORMAT = (TYPE = CSV)
```



COPY INTO VS INSERT

- Snowflake is optimized for bulk load and batched DML using the COPY INTO command
- Use COPY INTO to load data rather than INSERT with SELECT
- Use INSERT only if needed for transformations not supported by COPY INTO
- Batch INSERT statements
 - INSERT w/ SELECT
 - CREATE TABLE AS SELECT (CTAS)
 - Minimize frequent single row DMLs



QUERYING STAGE DATA

File format

```
177 select
178     substr($1, 0, 15) PTS,
179     substr($1, 16, 3) REC_TYPE,
180     substr($1, 19, 60) COMPANY_NAME,
181     substr($1, 79, 10) CIK,
182     IFF(substr($1, 16, 3) = 'CMP', substr($1, 89, 4), substr($1, 40, 4)) STATUS
183   from @TPCDI_FILES/tpcdi-5/Batch1/FINWIRE
184   (FILE_FORMAT => 'TXT_FIXED_WIDTH')
185   where substr($1, 16, 3) = 'CMP';
186
```

Several functions can be used

Results Data Preview

✓ [Query ID](#) [SQL](#) 2.46s 2,500 rows Add filters

Row	PTS	REC_TYPE	COMPANY_NAME	CIK	STATUS
1	19671209-013009	CMP	RSWvpXShAohbCnBYubXQUbMLtPavDHUQn	0000000102	ACTV
2	19671210-112810	CMP	VNUMGRITCSQKEfjLfhaUANwnlPAHDkoFxyHfyd...	0000000027	INAC
3	19671211-075534	CMP	FQCxtQRoMnVInjwEERVRrjOJDAlyWTEKGCbB...	0000000100	ACTV
4	19671215-123401	CMP	dBBCsCzeivSxrOWJZkMbNfLitfJSVqvvAy	0000000105	ACTV
5	19671216-191413	CMP	OKehKyGaHgbEeBcBDAnFaXwXmCfzDClig	0000000098	ACTV



LOAD/UNLOAD TRANSFORMATION DIFFERENCES

LOAD	UNLOAD
File formats: CSV, TSV, JSON, Avro, ORC, Parquet	CSV, TSV, JSON, Parquet only
No joins, filters, or aggregations in SELECTs	No limitations on SELECT features
Into TABLE from STAGE	Into STAGE from TABLE
Data goes into only a single table	Data can come from any number of tables



MONITORING

- Monitor the status of each COPY command run on the History tab page of the Snowflake UI

TABLE_NAME	LAST_LOAD_TIME	STATUS	ROW_COUNT	ROW_PARSED	FIRST_ERROR_MESSAGE
TEST	2018-11-20 07:44:22.868 -0800	LOAD_FAILED	0	3	Date '1,2,3' is not recognized

- Use the LOAD_HISTORY Information Schema view to retrieve the history of data loaded into tables using the COPY command

```
SELECT * FROM information_schema.load_history
WHERE SCHEMA_NAME=current_schema() AND
TABLE_NAME='my_table' AND
LAST_LOAD_TIME > 'Fri, 01 APR 2016 16:00:00 -800';
```



QUIZ

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION #1

Metadata is created when you load data from a file in an internal stage into a Snowflake table.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #1

Metadata is created when you load data from a file in an internal stage into a Snowflake table.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #2

A stage:

- A.** Is a cloud file repository
- B.** Simplifies and streamlines bulk loading and unloading
- C.** Can be internal or external
- D.** All of the above
- E.** None of the above



QUESTION #2

A stage:

- A. Is a cloud file repository**
- B. Simplifies and streamlines bulk loading and unloading**
- C. Can be internal or external**
- D. All of the above**
- E. None of the above**



QUESTION #3

A file format is a named object that (choose all that apply):

- A. Stores data from a loaded file in a specific, named format
- B. Is securable
- C. Stores information needed to parse files during load/unload



QUESTION #3

A file format is a named object that (choose all that apply):

- A. Stores data from a loaded file in a specific, named format
- B. Is securable
- C. Stores information needed to parse files during load/unload



QUESTION #4

Data loading recommendations include:

- A. Split large files into smaller files of 10MB – 100MB
- B. Organize data into logical paths that only contain the files to be loaded
- C. Use wildcards late in the file path definition, to reduce scanning
- D. The greater the number of files to load, the smaller the warehouse required



QUESTION #4

Data loading recommendations include:

- A. ~~Split large files into smaller files of 10MB – 100MB~~
- B. Organize data into logical paths that only contain the files to be loaded
- C. Use wildcards late in the file path definition, to reduce scanning
- D. ~~The greater the number of files to load, the smaller the warehouse required~~



QUESTION #5

You can unload to which of the following?

- A. Table stage
- B. Internal named stage
- C. External stage
- D. All of the above
- E. None of the above

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION #5

You can unload to which of the following?

- A. ~~Table stage~~
- B. ~~Internal named stage~~
- C. ~~External stage~~
- D. All of the above
- E. ~~None of the above~~

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION #6

COPY INTO @%[stage name] is for unloading into a named internal stage.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #6

COPY INTO @%[stage name] is for unloading into a named internal stage.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #7

When unloading from a SELECT statement into a stage, you can include joins or any other SQL command in the SQL statement.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #7

When unloading from a SELECT statement into a stage, you can include joins or any other SQL command in the SQL statement.

- A. True
- B. False



QUESTION #8

Snowpipe's charges are calculated per-second and per core.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION #8

Snowpipe's charges are calculated per-second and per core.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION #9

Via REST API, Snowpipe can only reference external stages as a source.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #9

Via REST API, Snowpipe can only reference external stages as a source.

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #10

The COPY command supports column reordering, column omission, and CAST using a SELECT statement

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION #10

The COPY command supports column reordering, column omission, and CAST using a SELECT statement

- A. True
- B. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



LAB EXERCISE: 6

Loading and Unloading Structured Data

30 minutes

Tasks:

- Create tables and file formats
- Load a file into a table using COPY INTO
- Describe a stage
- Load a GZIP compressed file
- Load data using the Load Data Wizard
- Unload data into a table stage using COPY INTO
- Unload data into an internal stage using COPY INTO



TASKS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Creating and Managing Tasks

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



CREATING AND MANAGING TASKS

Roger.Andre1@t-mobile.com 26.Oct.2021 @Snowflake 2020-do-not-copy



WORKING WITH TASKS

- Executes a DDL/DML/SQL statement or Stored Procedure
- Runs on a defined schedule, or after completion of a previous task
- Use Cases:
 - Keep aggregates up-to-date
 - Generate data for periodic reports
 - Copy data into or out of Snowflake



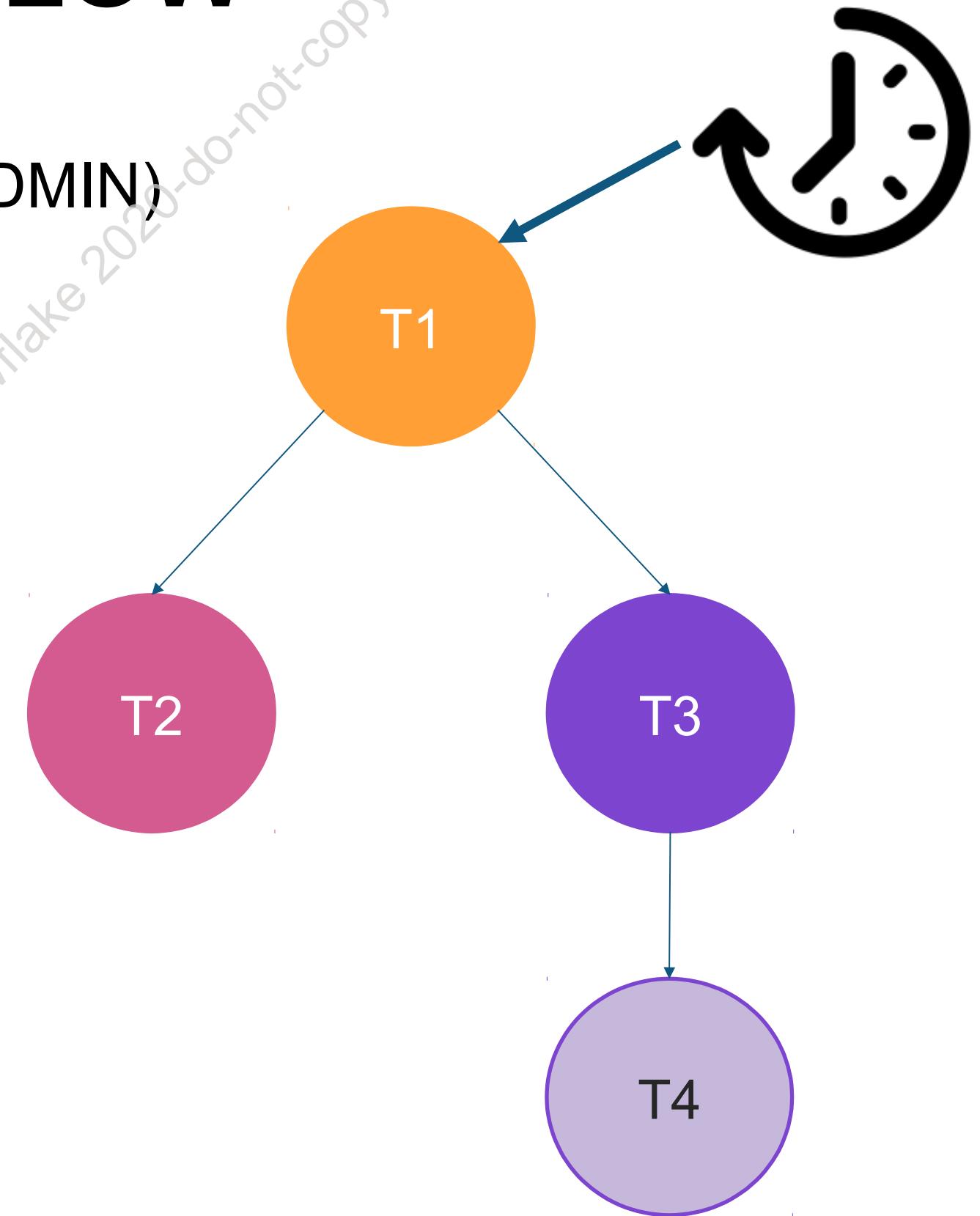
TASKS WORKFLOW

1. Create a Task Administrator Role (or use ACCOUNTADMIN)

2. CREATE TASK mytask ...

- Specify a warehouse
- Optionally Specify a schedule
- Optionally specify a condition
- Define the task

3. ALTER TASK mytask RESUME



TASK SYNTAX

CREATE OR REPLACE TASK <task name>

--Specify a warehouse, required

WAREHOUSE = <warehouse name>

--Specify a schedule - by minutes

SCHEDULE = '<integer> minute'

--Specify a schedule - using cron expression

SCHEDULE = 'USING CRON <expression>'

--Specify a condition

AFTER <task name>

AS

<sql statement> OR <stored procedure call>



SPECIFYING A SCHEDULE

- Schedule can be expressed in two ways:

1. '**<num>** minute'

- Task will run every **<num>** minutes.
- Example: '60 minute'.

2. USING CRON **<expression> <time_zone>**

- Specifies a cron expression and time zone for running the task.
- Supports a subset of standard cron utility syntax.



SPECIFYING A SCHEDULE (CONT'D)

BUILDING CRON EXPRESSIONS

- Cron expressions are built by specifying values for minute, hour, day, month and day of week
- Cron expressions must include a time zone (i.e., America/Chicago)
- Cron expressions can use the operators and parameter value ranges shown below:

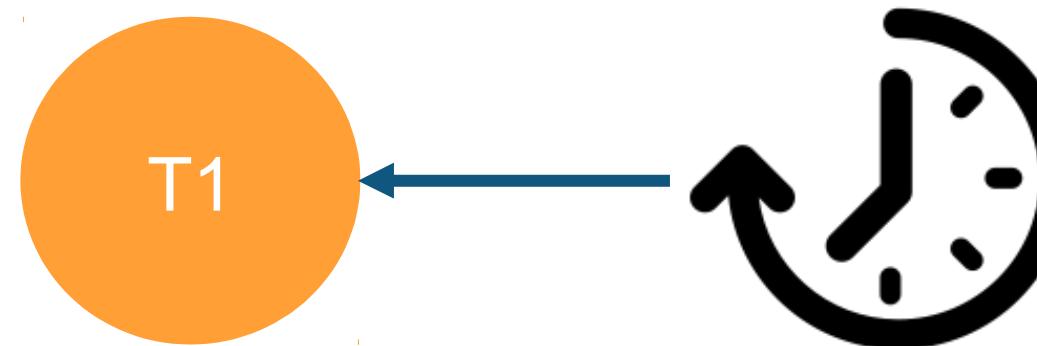
<u>Operators</u>	<u>Potential parameter values</u>
*	• minute (0-59)
,	• hour (0-23)
-	• day of month (1-31, or L)
/	• month (1-12, JAN-DEC)
	• day of week (0-6, SUN-SAT, or L)

Min	Hour	Day	Month	Day	Time Zone	Cron Expression Meaning
0	0	1	*	*	America/Chicago	At 00:00 am on the 1st of every month
1	0	*	*	SUN	America/Chicago	At 00:01 am every Sun, every month of the year
0	*	1	1,2,7,10	*	America/Chicago	At minute 0 on the 1 st of Jan, Apr, Jul, Oct
*	*	*	*	*	America/Chicago	Every minute of every hour of every day



SPECIFYING A CONDITION: USING AFTER

- For single tasks: A schedule **must** be defined for the task, or the task will never run.
- For a tree of tasks: A schedule **must** be defined for the root task, or the tree will never execute
- A schedule **cannot** be specified for child tasks in a tree of tasks. Use AFTER instead:



```
CREATE OR REPLACE TASK task1  
WAREHOUSE = 'WAREHOUSE1'  
SCHEDULE = 'USING CRON . . .'  
AS  
. . .
```

```
CREATE OR REPLACE TASK task1  
WAREHOUSE = 'WAREHOUSE1'  
SCHEDULE = 'USING CRON . . .'
```

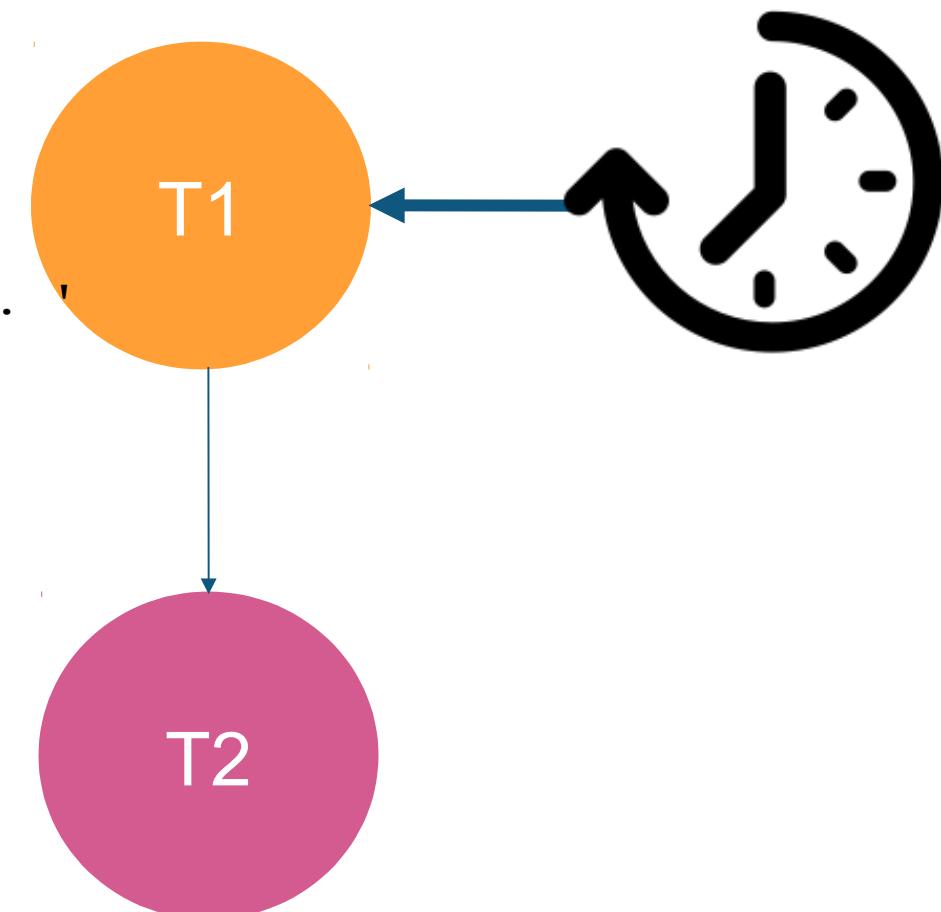
AS

. . .

```
CREATE OR REPLACE TASK task2  
WAREHOUSE = 'WAREHOUSE2'  
AFTER task1
```

AS

. . .



TASKS EXAMPLE #1

- Runs an ELT task every 60 minutes:

```
CREATE OR REPLACE TASK elt_employee
  WAREHOUSE = hr_wh
  SCHEDULE = '60 minute'
AS
  COPY INTO employee_raw_data FROM @my_stage;
```

```
CREATE OR REPLACE TASK elt_employee
  WAREHOUSE = hr_wh
  SCHEDULE = 'USING CRON 0 * * * * America/Chicago'
AS
  COPY INTO employee_raw_data FROM @my_stage;
```



TASKS EXAMPLE #2

- Simple tree that runs an ELT task every 60 minutes, then executes a child task:

```
CREATE OR REPLACE TASK elt_employee
  WAREHOUSE = hr_wh
  SCHEDULE = 'USING CRON 0 * * * * America/Chicago'
AS
  COPY INTO employee_raw_data FROM @my_stage;
```

```
CREATE OR REPLACE TASK process_employee
  WAREHOUSE = hr_wh
  AFTER elt_employee
AS
  INSERT INTO employees(first_name, last_name)
  SELECT firstname, lastname
  FROM employee_raw_data
  WHERE status = '1';
```



TASKS EXAMPLE #3

- Two tasks that increase a warehouse size at the start of the last two weeks of a fiscal quarter when the analytical load is high, then decrease it at the start of the next fiscal quarter:

```
CREATE OR REPLACE TASK acct_increase_wh_size
WAREHOUSE = acct_wh
SCHEDULE = 'USING CRON 0 0 15 3,6,9,12 * America/Chicago'
AS
ALTER WAREHOUSE acct_wh SET warehouse_size = '2x-large';
```

```
CREATE OR REPLACE TASK acct_decrease_wh_size
WAREHOUSE = acct_wh
SCHEDULE = 'USING CRON 0 0 1 1,4,7,10 * America/Chicago'
AS
ALTER WAREHOUSE acct_wh SET warehouse_size = 'medium';
```



QUIZ

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION #1

What does this cron expression express?

USING CRON 0 0-23 * * *

1. Every minute of every hour of every day
2. At 00:00 am on the 1st of every month
3. The first minute of every hour of every day
4. At 00:00 am on the 1st of every month



QUESTION #1

What does this cron expression express?

USING CRON 0 0-23 * * *

1. Every minute of every hour of every day
2. At 00:00 am on the 1st of every month
3. The first minute of every hour of every day
4. At 00:00 am on the 1st of every month



QUESTION #2

True or False: The expected average run time is the average difference between the scheduled and completed times for a task, excluding any time period in which the task was queued.

1. True
2. False



QUESTION #2

True or False: The expected average run time is the average difference between the scheduled and completed times for a task, excluding any time period in which the task was queued.

1. True
2. False

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake™ do-not-copy



QUESTION #3

Which of the following is true regarding a tree of tasks? Choose all that apply.

1. Root tasks must have a schedule, or the tree will never execute.
2. Child tasks can have schedules.
3. Child tasks cannot have schedules.
4. Child tasks can have an AFTER clause that triggers them to execute once a previous task has been completed.



QUESTION #3

Which of the following is true regarding a tree of tasks? Choose all that apply.

1. Root tasks must have a schedule, or the tree will never execute.
2. ~~Child tasks can have schedules.~~
3. Child tasks cannot have schedules.
4. Child tasks can have an AFTER clause that triggers them to execute once a previous task has been completed.



SNOWFLAKE FUNCTIONS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Snowflake Functions Overview
- User-Defined Functions
- Stored Procedures
- High Performance Functions

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SUPPORTED FUNCTION TYPES

Type	Description
Scalar	Takes 0 or more parameter values, returns a single value.
Aggregate	Performs a calculation on a set of values, and returns a single value.
Window	Aggregate functions that operate on a subset of rows within the input rows.
Table	Produces a collection of rows (either a nested table or a vararray) that can be queried like a physical database table. Use in the FROM clause of a query.
System	Used to execute action in the system, or return information about the system.
User-Defined	Written in JavaScript or SQL to modularize queries or portions of queries.



SCALAR FUNCTIONS

Performs a calculation on 0 or more parameters, and returns a single value

I_shipdate
1993-03-08
1996-05-15
1995-09-22

```
SELECT  
TO_VARCHAR(I_shipdate, 'mon dd, yyyy') AS shipdate  
FROM lineitem LIMIT 10;
```

shipdate
Mar 08, 1993
May 15, 1996
Sep 22, 1995



AGGREGATE FUNCTIONS

- Performs a calculation on a set of values, and returns a single value

Value
1
8
12
7

```
SELECT SUM(value) AS TOTAL  
FROM mytable;
```

TOTAL
28

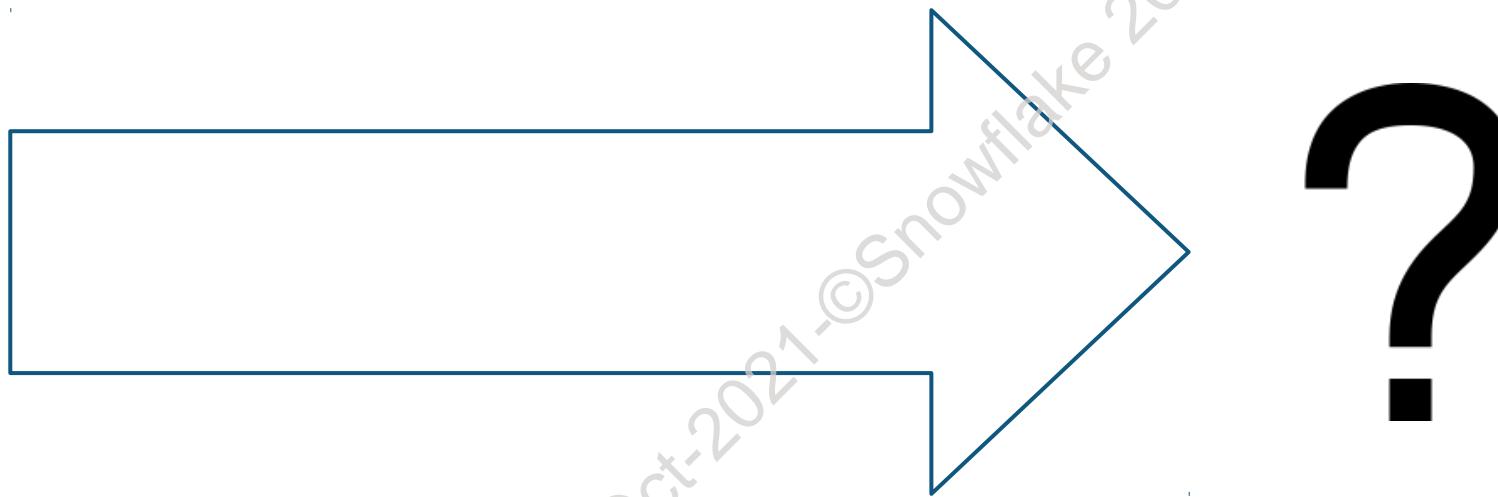
- Often used with the GROUP BY clause of the SELECT statement
- Except for COUNT, aggregate functions ignore null values



WINDOW FUNCTIONS

Aggregate functions that work on a subset (window) of the input rows

DATE	SALES
2019-08-01	8.73
2019-08-02	129.95
2019-08-03	13.75
2019-08-04	21.75
2019-08-05	115.87



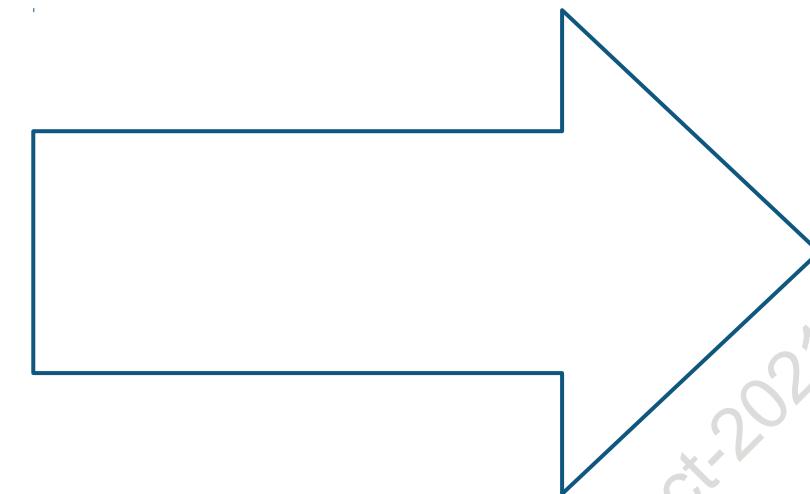
```
SELECT date, sales, SUM(sales)
OVER (ORDER BY date ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
AS mtd FROM aug_sales;
```



WINDOW FUNCTIONS

Aggregate functions that work on a subset (window) of the input rows

DATE	SALES
2019-08-01	8.73
2019-08-02	129.95
2019-08-03	13.75
2019-08-04	21.75
2019-08-05	115.87



DATE	SALES	MTD
2019-08-01	8.73	8.73
2019-08-02	129.95	138.68
2019-08-03	13.75	152.43
2019-08-04	21.75	174.18
2019-08-05	115.87	290.05

```
SELECT date, sales, SUM(sales)  
OVER (ORDER BY date ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
```

AS mtd

FROM aug_sales;



© 2021 Snowflake Computing Inc. All Rights Reserved

TABLE FUNCTIONS

- Take scalar expressions as input
- Return a set of rows instead of a single scalar value
- Appear in the FROM clause

```
SELECT event_timestamp AS time, user_name  
FROM TABLE (login_history_by_user())  
ORDER BY event_timestamp;
```

	TIME	USER_NAME
2021-02-12	08:11:00.166	-0700 DEBORAH
2021-02-12	09:14:08.128	-0600 DAVE
2021-02-12	08:09:17.004	-0800 JOON



SYSTEM FUNCTIONS

CONTROL

- ABORT_SESSION
- ABORT_TRANSACTION
- CANCEL_QUERY
- CANCEL_ALL_QUERIES
- PIPE_FORCE_RESUME
- TASK_DEPENDENTS_ENABLE
- WAIT

INFORMATION

- CLUSTERING_DEPTH
- CLUSTERING_INFORMATION
- CURRENT_USER_TASK_NAME
- PIPE_STATUS
- STREAM_HAS_DATA
- WHITELIST

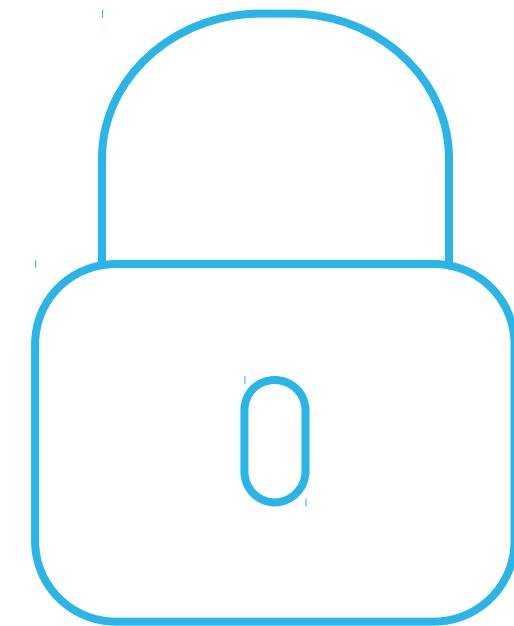
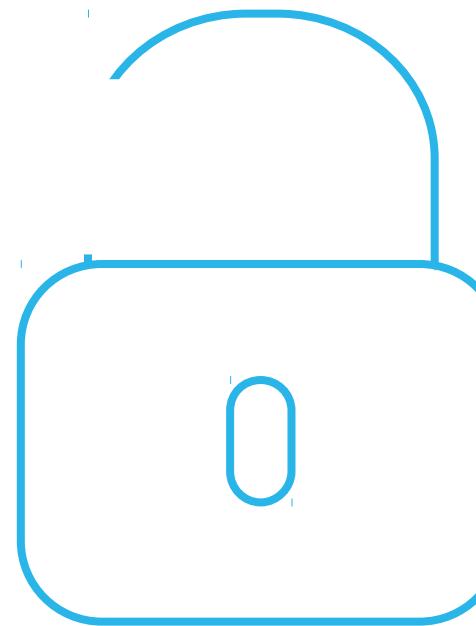


USER-DEFINED FUNCTIONS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



USER-DEFINED FUNCTIONS



SQL | JAVASCRIPT

- Perform custom operations that are not available through the built-in functions
- SQL and JavaScript supported
- No DDL/DML support
- Can be unsecure or secure
- Return a singular scalar value or, if defined as a table function, a set of rows



SQL UDF EXAMPLE

```
CREATE OR REPLACE FUNCTION order_cnt(custkey number(38,0))
RETURNS number(38,0)
AS
$$
SELECT COUNT(1)
FROM "SNOWFLAKE_SAMPLE_DATA"."TPCH_SF1"."ORDERS"
WHERE o_custkey = custkey
$$;

SELECT C_name, C_address, order_cnt(C_custkey)
FROM "SNOWFLAKE_SAMPLE_DATA"."TPCH_SF1"."CUSTOMER"
LIMIT 10;
```



JAVASCRIPT UDF

```
CREATE OR REPLACE FUNCTION convert2fahrenheit(kelvin double)
RETURNS double
LANGUAGE JAVASCRIPT
AS
$$
    return (KELVIN * 9/5 - 459.67);
$$;

SELECT convert2fahrenheit(290);
```

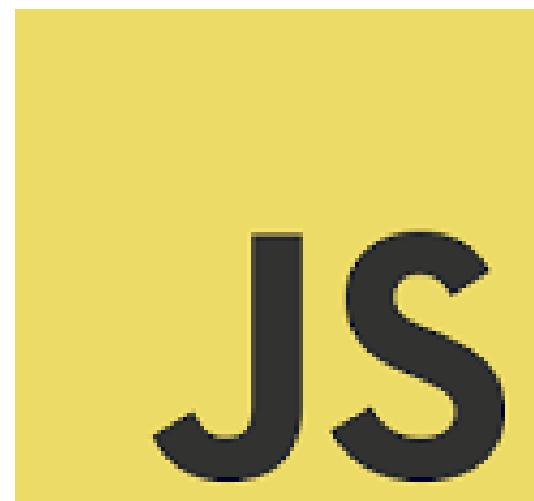


STORED PROCEDURES

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



STORED PROCEDURES



JAVASCRIPT

- Allow procedural logic and error handling that straight SQL does not support
- Implemented through JavaScript and, optionally (commonly), SQL
- JavaScript provides the control structures
- SQL is executed within the JavaScript by calling functions in an API
- Argument names are:
 - Case-insensitive in the SQL portion of stored procedure code
 - Case-sensitive in the JavaScript portion



COMPONENTS OF STORED PROCEDURE

```
CREATE OR REPLACE PROCEDURE STPROC1(FLOAT_PARAM1 FLOAT)
RETURNS STRING
LANGUAGE JAVASCRIPT STRICT
AS
$$ // dollar signs mark beginning and end of the Javascript code

try {
    snowflake.execute (
        {sqlText: "INSERT INTO STPROC_TEST_TABLE1 (NUM_COL1) VALUES
            (" + FLOAT_PARAM1 + ")"}
    );
    return "Succeeded." // Return status
} catch (err) {
    return "Failed: " + err; // status
}
$$ ;
CALL stproc1(5.14::FLOAT);
```



COMPONENTS OF STORED PROCEDURE

Data type to return

```
CREATE PROCEDURE stproc1(float_param1 FLOAT)
RETURNS STRING
LANGUAGE JAVASCRIPT
STRICT
AS
$$ // marks beginning and end of the code
try {
    snowflake.execute (
        {sqlText: "INSERT INTO stproc_test_table1 (num_col1)
VALUES (" +FLOAT_PARAM1 + ")"});
    return "Succeeded." // Return status
} catch (err) {
    return "Failed: " + err; // status
}
$$;

call stproc1(5.14::FLOAT);
```



COMPONENTS OF STORED PROCEDURE

Language (currently only
JavaScript supported)



```
CREATE PROCEDURE stproc1(float_param1 FLOAT)
RETURNS STRING
LANGUAGE JAVASCRIPT
STRICT
AS
$$ // marks beginning and end of the code
try {
    snowflake.execute (
        {sqlText: "INSERT INTO stproc_test_table1 (num_col1)
VALUES (" +FLOAT_PARAM1 + ")"});
    return "Succeeded." // Return status
} catch (err) {
    return "Failed: " + err; // status
}
$$;

call stproc1(5.14::FLOAT);
```



COMPONENTS OF STORED PROCEDURE

SQL statements to execute

```
CREATE PROCEDURE stproc1(float_param1 FLOAT)
RETURNS STRING
LANGUAGE JAVASCRIPT
STRICT
AS
$$ // marks beginning and end of the code
try {
    snowflake.execute (
        {sqlText: "INSERT INTO stproc_test_table1 (num_col1)
VALUES (" +FLOAT_PARAM1 + ")"});
    return "Succeeded." // Return status
} catch (err) {
    return "Failed: " + err; // status
}
$$;

call stproc1(5.14::FLOAT);
```



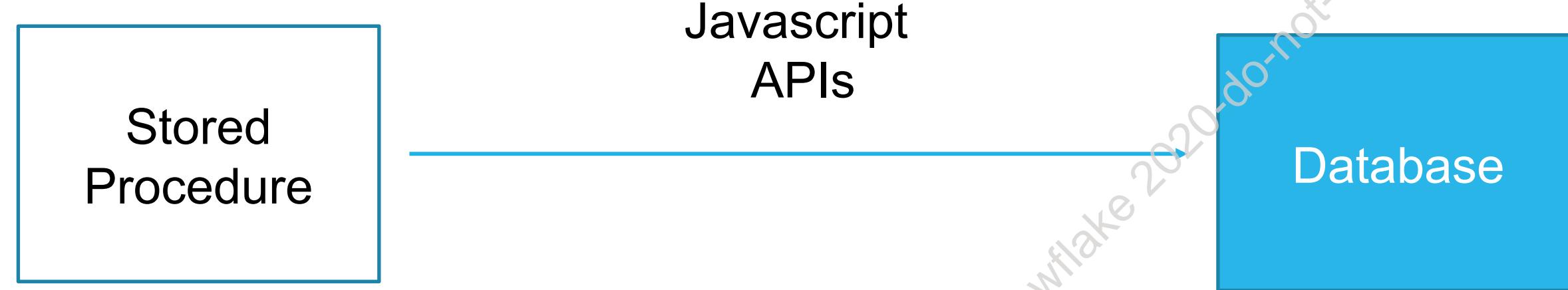
COMPONENTS OF STORED PROCEDURE

```
CREATE PROCEDURE stproc1(float_param1 FLOAT)
RETURNS STRING
LANGUAGE JAVASCRIPT
STRICT
AS
$$ // marks beginning and end of the code
try {
    snowflake.execute (
        {sqlText: "INSERT INTO stproc_test_table1 (num_col1)
VALUES (" +FLOAT_PARAM1 + ")"});
    return "Succeeded." // Return status
} catch (err) {
    return "Failed: " + err; // status
}
$$;
```

Invoke the stored procedure → **call stproc1(5.14::FLOAT);**



HOW IT WORKS



- Snowflake-provided JavaScript APIs
 - JavaScript objects and methods
 - Error handling and procedural logic
- SQL and database access/operations
 - Embed SQL code in the JavaScript
 - The SQL code executes database operations
 - Migrate other databases' stored procedure (SQL) code by embedding the SQL in JavaScript



STORED PROCEDURE SQL

EXECUTED THROUGH API OBJECTS

Object Class	Description
snowflake Object	Contains the methods in the stored procedure API. Accessible by default to JavaScript code (you do not need to create the object).
Statement Object	Provides the methods for executing a query statement, and accessing metadata (for example, information about columns and rows) about the statement
ResultSet Object	Contains the results returned by a query (zero or more rows, each with one or more columns). You iterate through a result set by repeating next() and taking some action
SfDate Object	JavaScript does not have a native data type that corresponds to Snowflake SQL TIMESTAMP data types. Used when you want to retrieve a TIMESTAMP and stored it as a variable.



STORED PROCEDURES VS UDFS

STORED PROCEDURE

- MAY return a value
- CAN NOT return a set of rows
- Can access database objects and issue SQL statements
- Can run procedure as owner or caller
 - Owner is default
 - Specified at creation time
- Stored Procedures support transactions.
 - The transactions must start and finish inside the stored procedure.

USER-DEFINED FUNCTION

- MUST return a value
- CAN return a set of rows (table)
- DDL and DML operations not permitted
- Runs as the function owner



HIGH-PERFORMING FUNCTIONS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



APPROXIMATION

- Snowflake uses *HyperLogLog* for a set of aggregate functions to estimate the approximate number of distinct values in a data set.
- *HyperLogLog* is a state-of-the-art cardinality estimation algorithm, capable of estimating distinct cardinalities of trillions of rows with an average relative error of a few percent.

```
SELECT COUNT(cust_orders),  
COUNT(distinct cust_orders),  
APPROX_COUNT_DISTINCT(cust_orders)  
FROM customers;
```

Count (cust_orders)	Count (distinct cust_orders)	HLL (cust_orders)
2205	2205	2220



COUNTING LARGE DISTINCT DATASETS

COUNT (DISTINCT col1)

- Slower
- Scalable
- Exact

```
SELECT COUNT(DISTINCT o_orderkey)  
FROM orders;
```

```
COUNT(DISTINCT O_ORDERKEY)
```

150000000

APPROX_COUNT_DISTINCT (col1)

- Much faster
- Approximate but within ~1.62% error
- Less memory intensive for column with a large number of distinct values

```
APPROX_COUNT_DISTINCT(o_orderkey)  
FROM orders;
```

```
APPROX_COUNT_DISTINCT(O_ORDERKEY)
```

145660677



ESTIMATION VS EXACT

MEDIAN (col1)

- Slower
- Scalable
- Exact

```
35 select median(ss_sales_price), ss_store_sk  
36 from snowflake_sample_data.tpcds_sf10tcl.store_sales  
37 group by ss_store_sk;
```

Results Data Preview

✓ Query ID SQL 18m59s

APPROX_PERCENTILE (col1)

- Much faster than MEDIAN
- Uses a constant amount of space regardless of the size of the input

```
42 select approx_percentile(ss_sales_price, 0.5), ss_store_sk  
43 from snowflake_sample_data.tpcds_sf10tcl.store_sales  
44 group by ss_store_sk;  
45  
46 select median(ss_sales_price)
```

Results Data Preview

✓ Query ID SQL 3m2s



FREQUENCY ESTIMATION

APPROX_TOP_K (<expr> [, <k> [, <counters>]])

Returns estimated frequency of most frequent values

APPROX_TOP_K_ACCUMULATE (<expr> , <counters>)

Skips the final estimation step and returns the Space-Saving state at the end of an aggregation

APPROX_TOP_K_COMBINE (<state> [, <counters>])

Combines (i.e., merges) input states into a single output state

APPROX_TOP_K_ESTIMATE (<state> [, <k>])

Computes a cardinality estimate of a Space-Saving state produced by

APPROX_TOP_K_ACCUMULATE and APPROX_TOP_K_COMBINE



SIMILARITY ESTIMATION

MINHASH(<k> , [distinct] expr+)

Estimates the approximate similarity between two or more data sets.

MINHASH_COMBINE([distinct] <state>])

Combines input MinHash states into a single MinHash output state.

This Minhash state can then be input to the APPROXIMATE_SIMILARITY function to estimate the similarity with other MinHash states.

APPROXIMATE_SIMILARITY ([distinct] <expr> [, . . .])

Returns an estimation of the similarity (Jaccard index) of inputs based on their MinHash states.

Returns a value between 0.0 (no similarity) to 1.0 (identical).

Equivalent to APPROXIMATE_JACCARD_INDEX.



MANAGING SECURITY

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Security Overview
- Access
- Authentication
- Authorization
- Data Protection
- Infrastructure

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



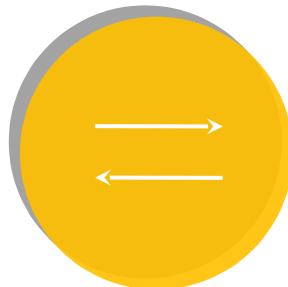
SECURITY OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy



SECURITY AT A GLANCE

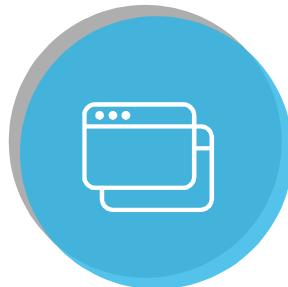
ALL EDITIONS COMBINED



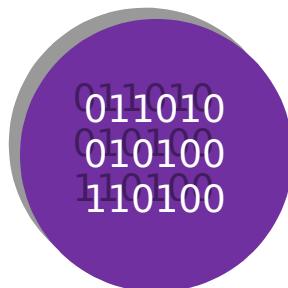
Access



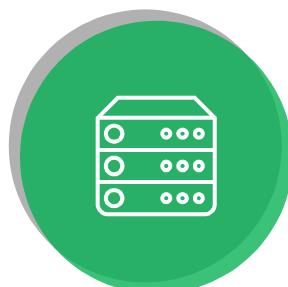
Authentication



Authorization



Data Protection



Infrastructure

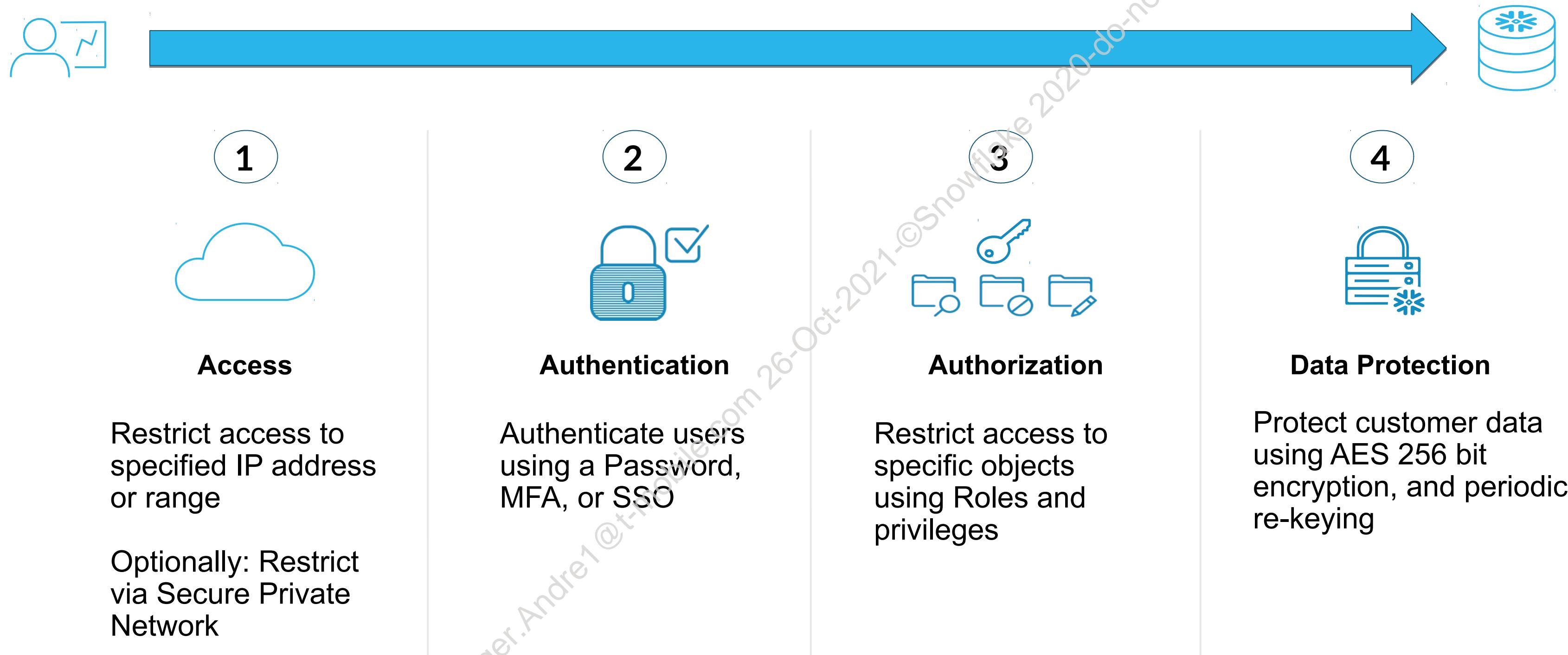


Snowflake operational controls

- NIST 800-53
- SOC2 Type 2
- SOC1 Type II
- ISO/IEC 27001:2013
- HIPAA
- PCI
- FedRAMP



SNOWFLAKE LAYERED SECURITY



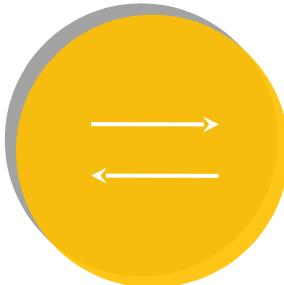
ACCESS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

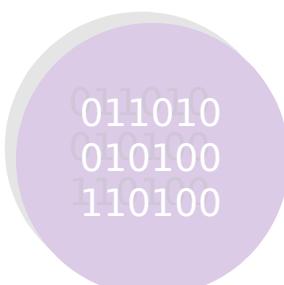


ACCESS

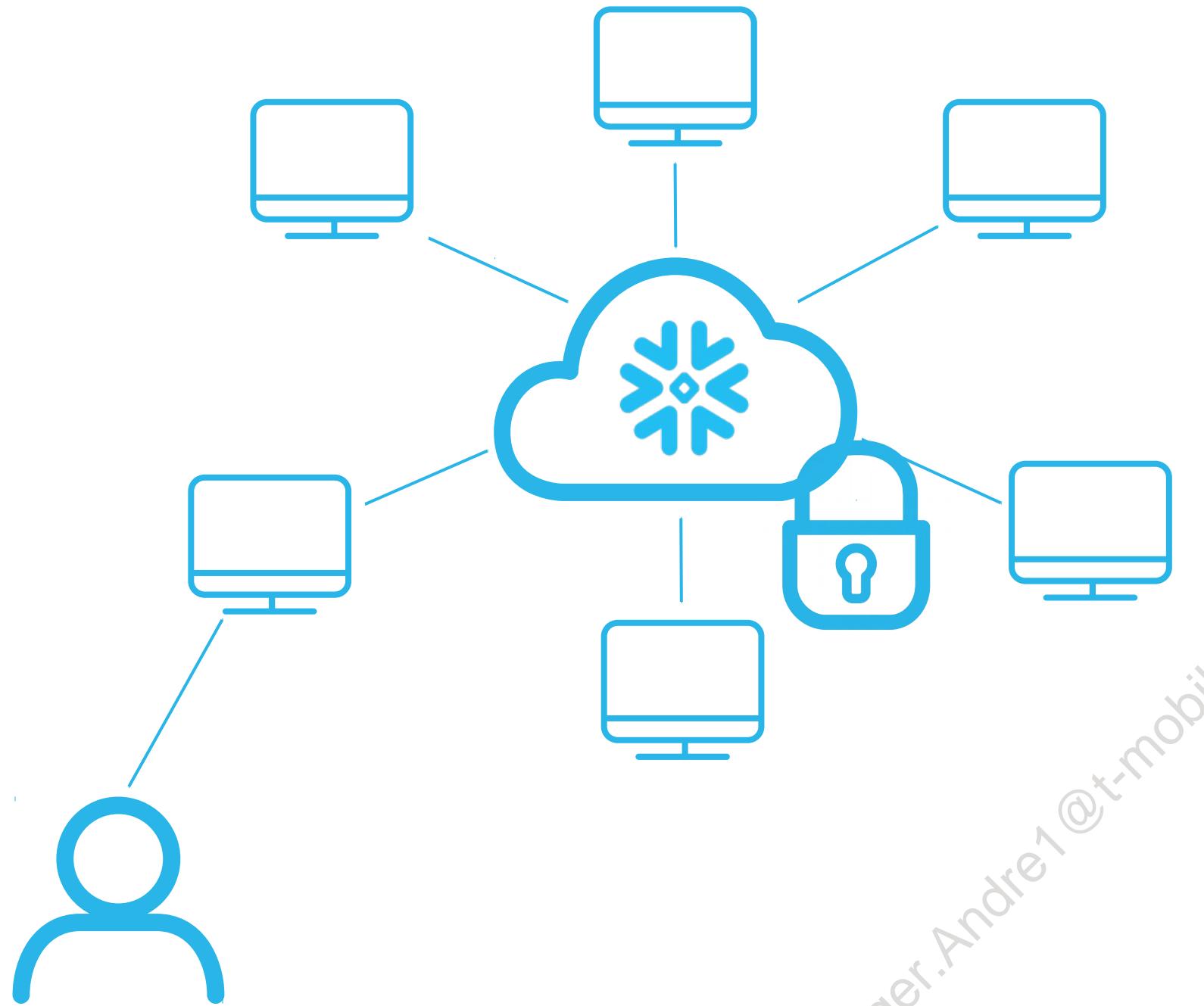
CAN YOU GET TO THE SNOWFLAKE ACCOUNT?



- IP lists allow and disallow
- TLS 1.2 encryption for all client communications
- Support for cloud-provider private network solutions



NETWORK SECURITY



- Network Policy support for IP lists allow and disallow
- User-Level Network Policy
 - Assign Network Policy to individual User
- Online certificate status protocol (OCSP) x.509 validation native to all Snowflake connectors
- Support for cloud provider connectivity options, such as the following:
 - PrivateLink (AWS and Azure) - no traffic over the public internet
 - DirectConnect (AWS) - dedicated network connection on-premise to AWS
 - Google options on roadmap



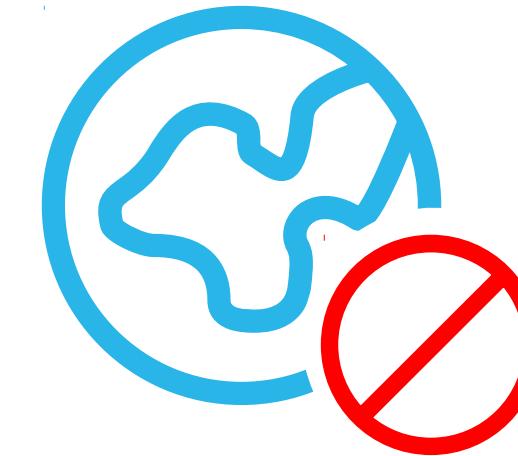
ACCOUNT-LEVEL NETWORK POLICY

- Restrict access to a Snowflake account to authorized network points (IPv4)
- Allow or Disallow (Blocked)
- Use CIDR notation to specify a subnet range
- One active network policy per account



Allowed IPs (Required)

IPv4 addresses or ranges **allowed** access to the Snowflake account



Blocked IPs (Optional)

IPv4 addresses or ranges **denied** access to the Snowflake account

CONFIGURE NETWORK POLICY

- User must be ACCOUNTADMIN or SECURITYADMIN
- ALLOWED_IP_LIST is required, BLOCKED_IP_LIST is optional

CREATE NETWORK POLICY <policy name>

ALLOWED_IP_LIST = (<IP address or range>, ...)

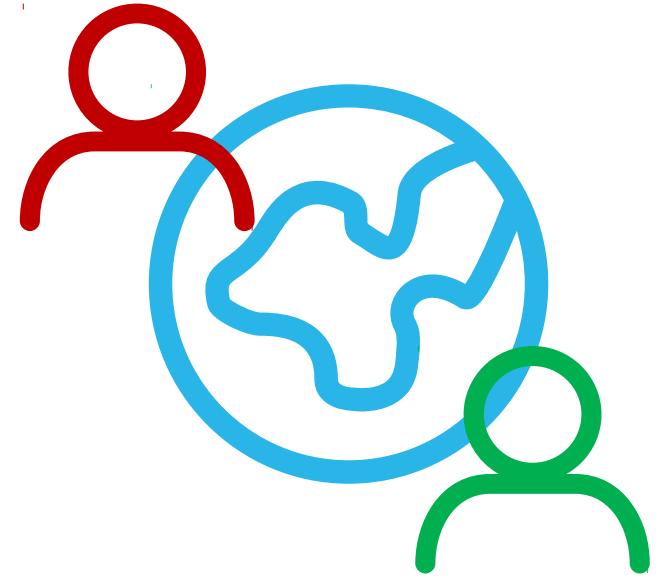
BLOCKED_IP_LIST = (<IP address(es) in allowed range>)

ALTER ACCOUNT SET NETWORK_POLICY = <policy name>



USER-LEVEL NETWORK POLICY

- Restricts access by user based on allowed/blocked IP address list
- User denied access if they attempt to log in from a restricted IP address
- Only one network policy can be associated with a user at any one time
- If a policy is assigned to a user who is already signed in, they are prevented from executing further queries
- There is no ability to create or edit a user level network policy using the graphical interface features of the WebUI
- However, you can execute the necessary SQL commands from a worksheet in the WEBUI.
Example:
 - **ALTER USER <name> SET NETWORK_POLICY = <string>**



ENSURING CONNECTIVITY

- Allow the following ports:
 - 443: Required for general Snowflake traffic
 - 80: Required for OCSP cache server, which listens for all Snowflake client communication
- Hostname Allow:
 - All Snowflake clients require access to cloud storage
 - Allow hostname patterns for the required hosts (see documentation for details)
 - Use SYSTEM\$WHITELIST() to obtain required hostnames for your Snowflake account



ACCESS FLOW STEPS

CONNECTION REQUEST



**Enforce network
policy set on the
account**

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

ACCESS FLOW STEPS

CONNECTION REQUEST



1.
**Enforce network
policy set on the
account**



2.
**Verify that the
account is not
locked / disabled**



ACCESS FLOW STEPS

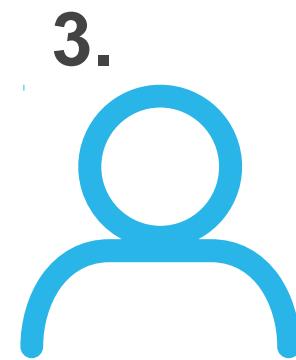
CONNECTION REQUEST



Enforce network policy set on the account



Verify that the account is not locked / disabled



Verify that the user is not locked / disabled

ACCESS FLOW STEPS

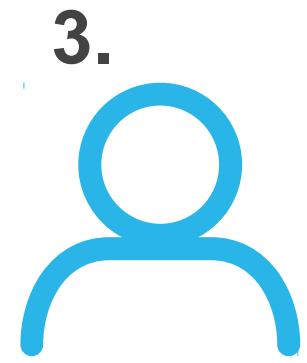
CONNECTION REQUEST



Enforce network policy set on the account



Verify that the account is not locked / disabled



Verify that the user is not locked / disabled



Resolve the account + user from the given request

ACCESS FLOW STEPS

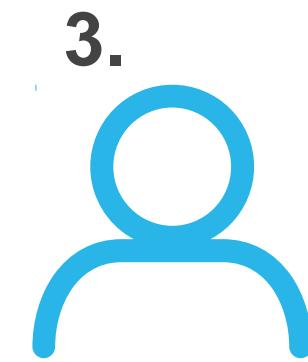
CONNECTION REQUEST



Enforce network policy set on the account



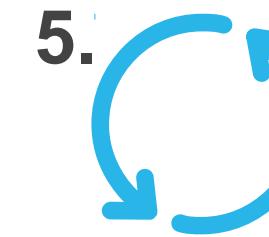
Verify that the account is not locked / disabled



Verify that the user is not locked / disabled



Resolve the account + user from the given request



Perform basic authentication checks

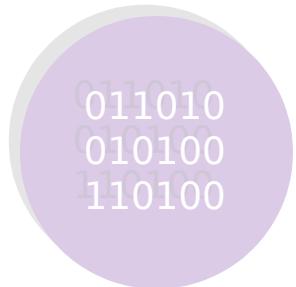
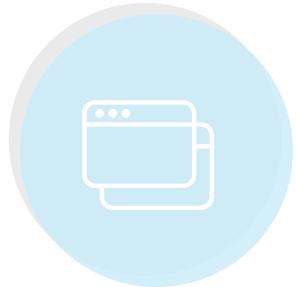
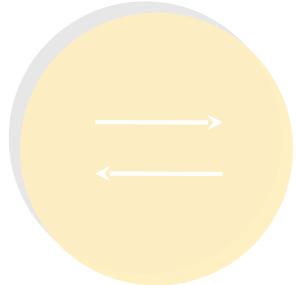
AUTHENTICATION

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



AUTHENTICATION

WHO ARE YOU?

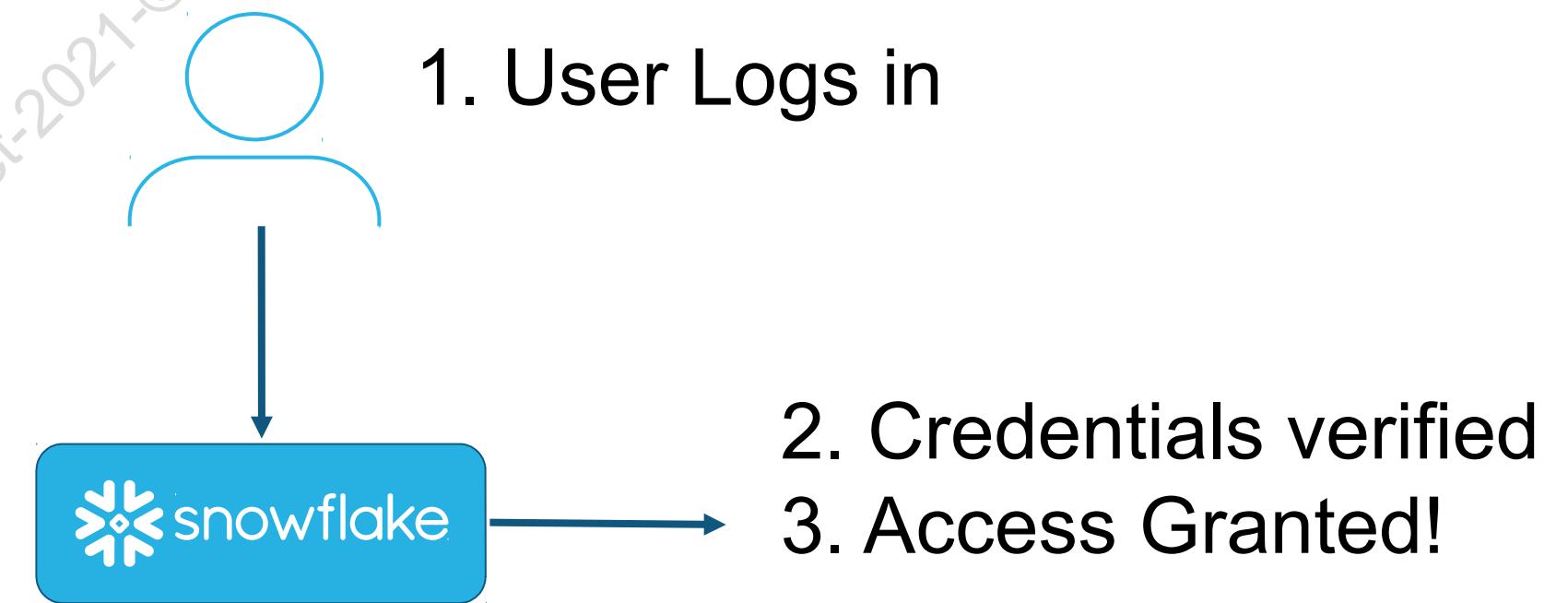


- Username and password
- Multi-factor authentication
- Federated authentication (SAML 2.0)
- Other authentication methods (OAuth, Key-Pair, SCIM)

Roger.Andre1@t-mobile.com 26 Oct 2021 ©Snowflake 2020-do-not-copy

USERNAME AND PASSWORD

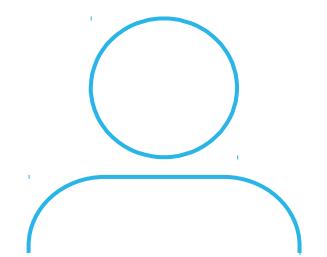
- Verify the password given matches the one associated with the user
- Default method
- Requirements:
 - 8-256 characters
 - At least one upper-case letter
 - At least one lower-case letter
 - At least one number



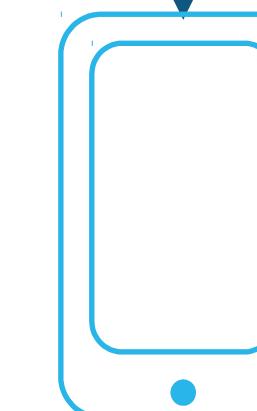
MULTI-FACTOR AUTHENTICATION

Authentication requires an extra step, after username/password is provided

1. User enables MFA
2. User Logs in



3. Request to Duo



5. Confirm on smartphone, or enter passcode

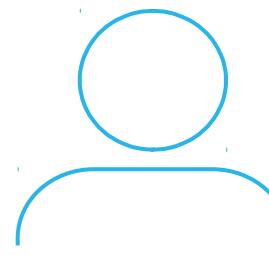
Roger.Andrei@t-mobile.com 26.01.2021 ©Snowflake 2020-Do-not-copy

6. Access Granted!

FEDERATED AUTHENTICATION (SSO)

Authenticate through an external identity provider, such as Okta

1. User Logs in



2. Request to Identity Provider (IdP)



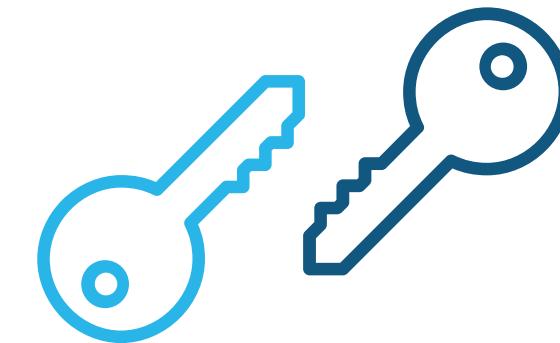
3. IdP authenticates User

5. Access Granted!

4. IdP sends confirmation

OTHER AUTHENTICATION METHODS

- **OAuth:** Authorizations on behalf of a user from a 3rd party service or application
- **KeyPair:** JSON Web Token (JWTs) signed using a public/private key pair with RSA encryption
- **SCIM** (System Cross-domain Identity Management): Automated management of users and roles in cloud applications



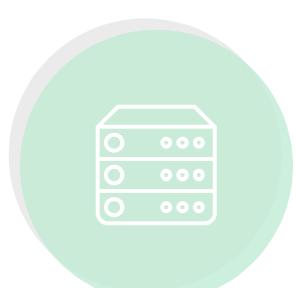
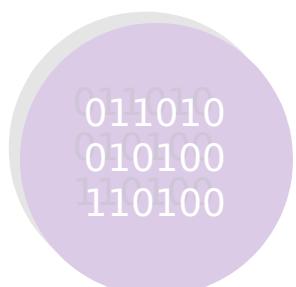
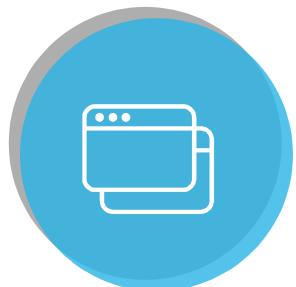
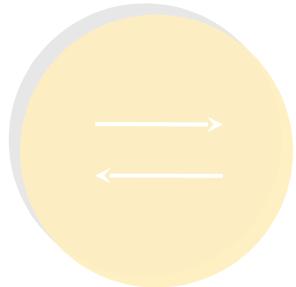
AUTHORIZATION

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



AUTHORIZATION

WHAT CAN YOU DO?



- Flexible and granular authorization
- Secure views and UDFs to protect information access
- Row-level access control

Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

AUTHORIZATION CONTROL

- Combination of Role-Based (RBAC) and Discretionary Control
 - Role-based: Access privileges are assigned to roles
 - Discretionary: Each object has an owner, which can grant others access to that object
- Privileges that can be set depend on the securable object
- Privileges can be broad or granular



SECURE VIEWS AND UDFs

- Prevent data from being indirectly exposed through programmatic methods
- Only authorized users can see the view or UDF definition
- Secure view and secure UDFs bypass some query optimizations to achieve greater security



ROW-LEVEL AUTHORIZATION

- Context functions can be used to provide row-level security
 - CURRENT_USER() – Returns the current users login name. Useful to limit personal information.
 - CURRENT_ROLE() – Returns the current role for the session.
 - CURRENT_ACCOUNT() – Returns the current account. Used when sharing data.
- This is commonly used with secured views to limit row access in a table.
- This example will only show the payroll information for the current user.

```
SELECT * FROM payroll  
JOIN employees ON payroll.empID = employees.empID  
AND employees.username = CURRENT_USER();
```



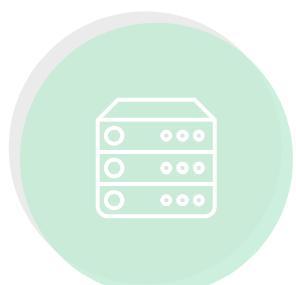
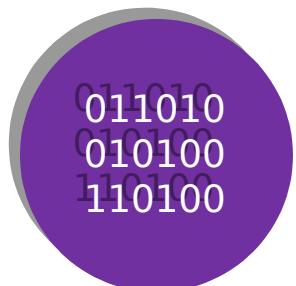
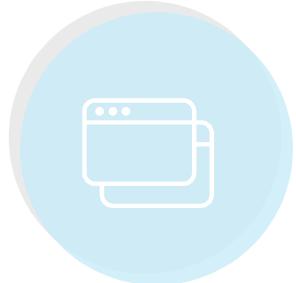
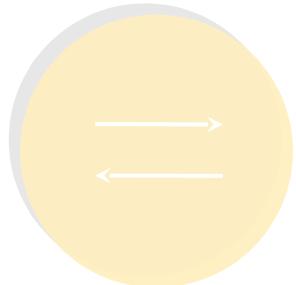
DATA PROTECTION

Roger.Andre1@t-mobile.com 26-Oct-2021 © Snowflake 2020-do-not-copy



DATA PROTECTION

IS YOUR DATA SAFE?



- Encrypted at rest and in motion
- Replication across availability zones
- Time Travel, Failsafe, and Replication Protection

Roger.Andre1@t-mobile.com 26-Oct-2020 ©Snowflake 2020-do-not-copy

ENCRYPTION

- Encryption at rest and in motion

- Hierarchical key model
 - Automatic key rotation
 - Periodic rekeying (Enterprise edition and above)

```
ALTER ACCOUNT SET PERIODIC_DATA_REKEYING = true;
```

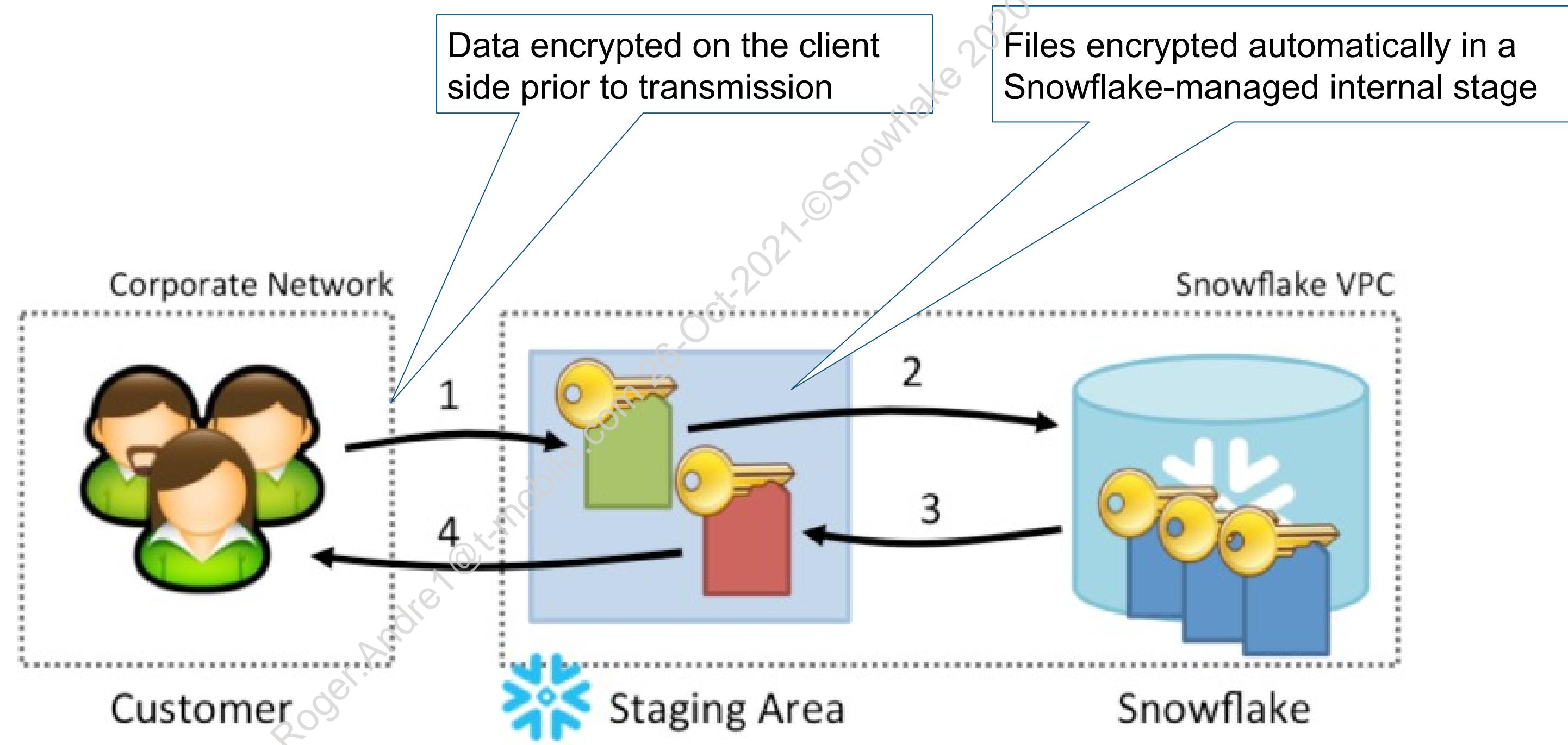
- Tri-Secret Secure (BYOK)

- Composite master key: combine customer key with a Snowflake-maintained key



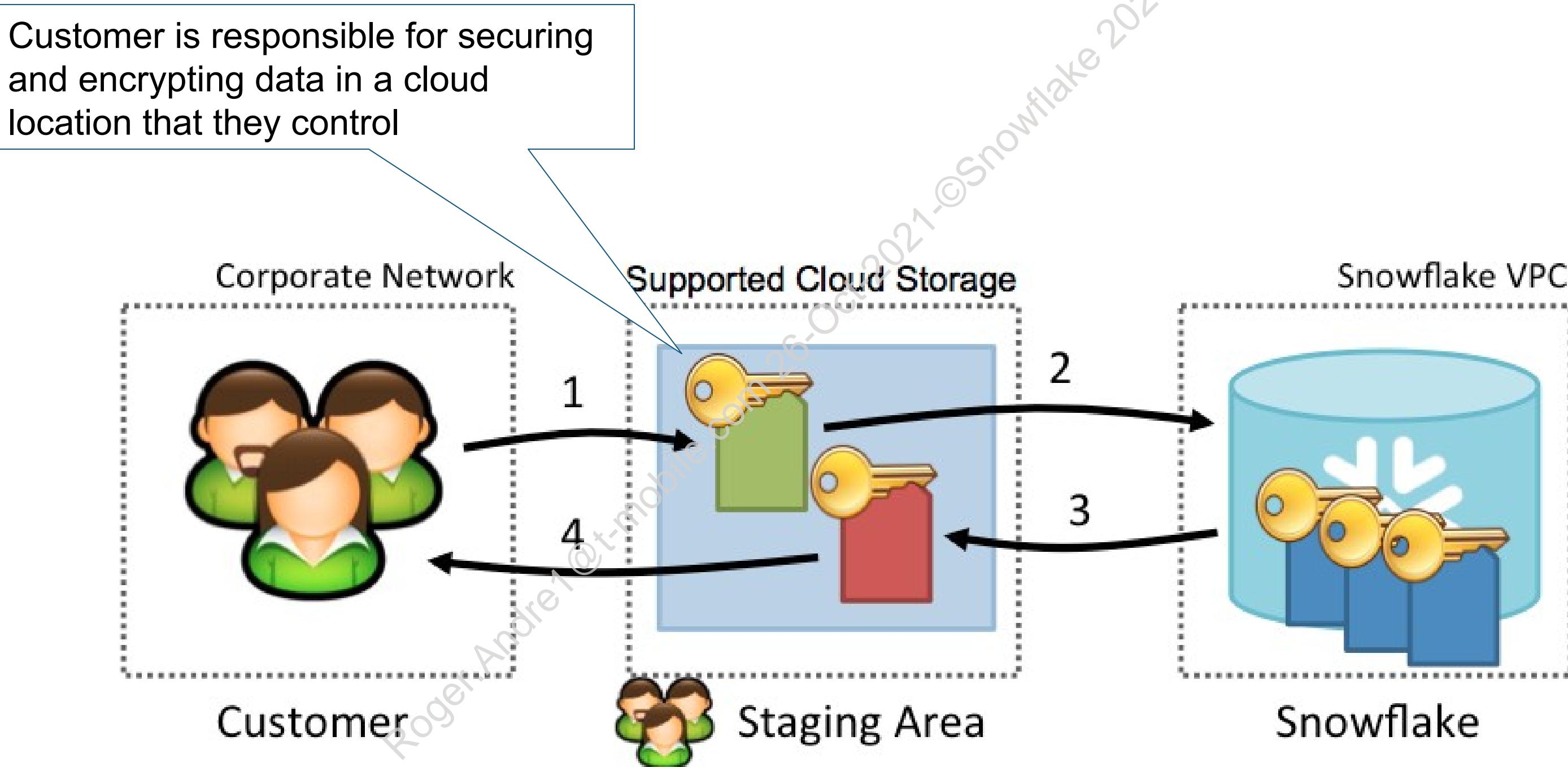
SNOWFLAKE-PROVIDED STAGING AREA

AUTOMATIC E2EE

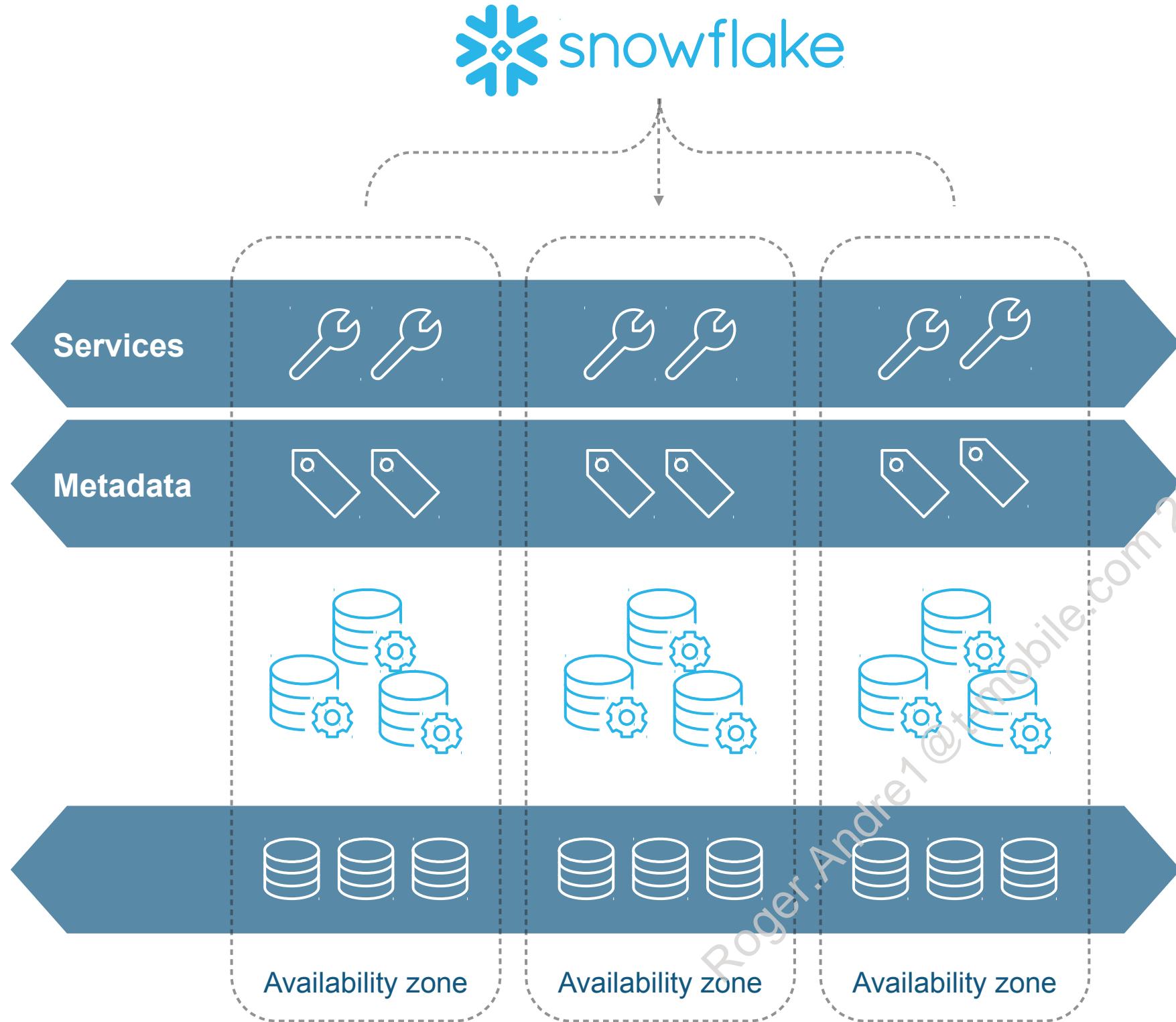


CUSTOMER-PROVIDED STAGING

ENCRYPTION/SECURITY MANAGED BY CUSTOMER



BUILT-IN CONTINUOUS AVAILABILITY



- Transparently synchronizes data across availability zones
 - Geographic separation
 - Separate power grids
- Fully online updates and patches
- Time Travel and Failsafe protection



INFRASTRUCTURE

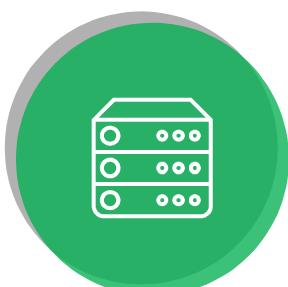
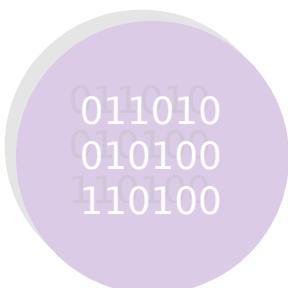
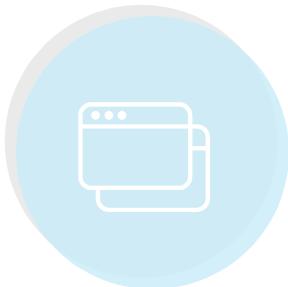
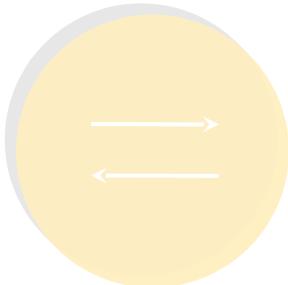
Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



INFRASTRUCTURE

IS YOUR INFRASTRUCTURE SECURE?

- Cloud provider's physical security
- Cloud provider's redundancy
- Limitless elasticity
- Regional data centers - US, EU, AP



ACCESS CONTROL AND USER MANAGEMENT

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Concepts
- System Roles
- Custom Roles and Inheritance
- Ownership
- Configure and Manage Access
- Recap

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



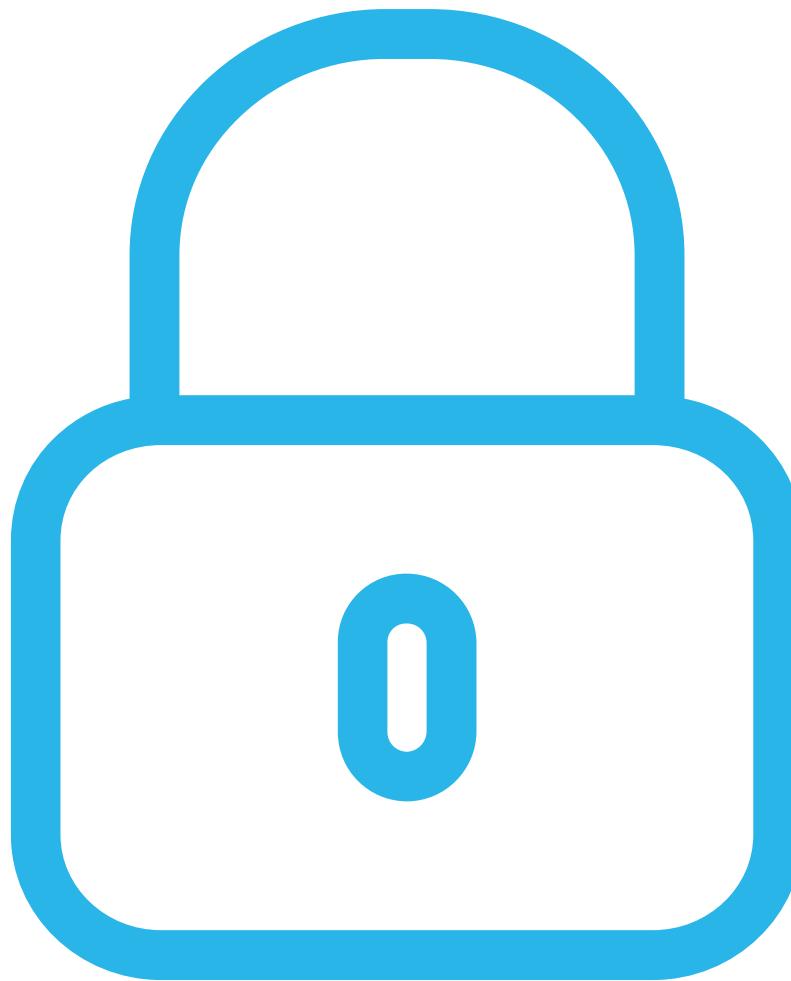
CONCEPTS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



ACCESS CONTROL

What can you do?



- Access Control is part of Snowflake's authorization model
- Access Control defines:
 - Who can use which roles
 - Who can access which objects
 - What operations can be performed on those objects
 - Who can create or alter access control policies



USERS AND ROLES

USERS

 SREEDHAR

 ANNA

 EVAN

 JENN

ROLES

PROD



TESTDEV

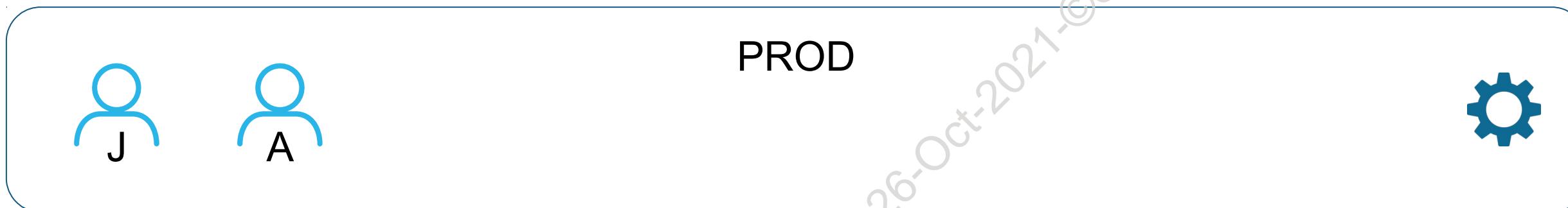


ANALYST



PRIVILEGES AND GRANTS

- A privilege is the right to do something on or with a Snowflake securable object
- A grant gives specific privileges to a role, or gives a user the right to use a role



```
GRANT ROLE PROD TO USER JENN;  
GRANT ROLE PROD TO USER ANNA;  
GRANT USAGE OF WAREHOUSE prod_wh TO ROLE PROD;
```

```
GRANT CREATE SCHEMA ON DATABASE prod_db TO ROLE PROD;  
privilege
```

grant



GRANT ROLES TO USERS

- Grant roles to a user, to enable the user to access (use) those roles

GRANT ROLE <name> TO USER <name>;

- Examples:

```
GRANT ROLE dba TO USER barney;
```

```
GRANT ROLE analyst, dba TO USER gail;
```



GRANT PRIVILEGES TO ROLES

- Grant privileges on securable objects to roles

```
GRANT <privilege(s)> ON <securable object(s)> TO ROLE <role>;
```

- Examples:

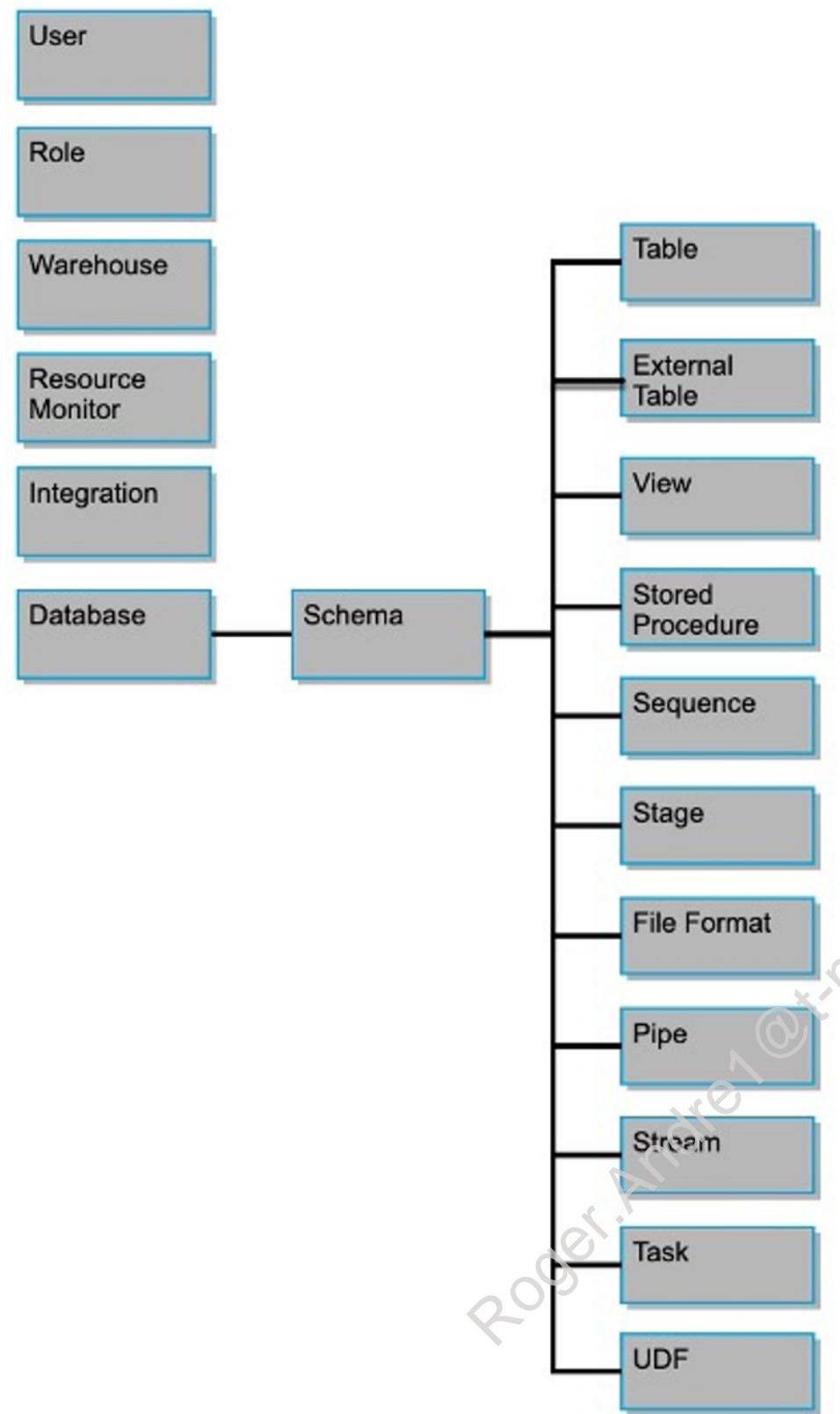
```
GRANT USAGE ON WAREHOUSE my_wh TO ROLE developers;
```

```
GRANT SELECT, INSERT, DELETE ON ALL TABLES IN SCHEMA my_schema  
TO ROLE analyst;
```



SECURABLE OBJECTS

Account



- Securable objects are constructs within Snowflake
- You grant privileges on securable objects
- Available privileges depend on the object type



AVAILABLE PRIVILEGES DEPEND ON THE OBJECT



CREATE ROLE | USER | WAREHOUSE | DATABASE, MANAGE GRANTS, MONITOR USAGE... **ON ACCOUNT**

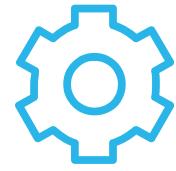
Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-don't-copy



AVAILABLE PRIVILEGES DEPEND ON THE OBJECT



CREATE ROLE | USER | WAREHOUSE | DATABASE, MANAGE GRANTS, MONITOR USAGE...**ON ACCOUNT**



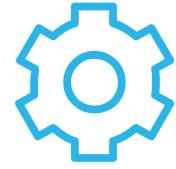
GRANT USAGE | MONITOR | MODIFY | OPERATE **ON WAREHOUSE**



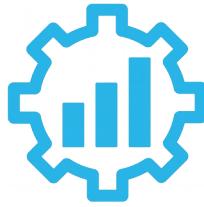
AVAILABLE PRIVILEGES DEPEND ON THE OBJECT



CREATE ROLE | USER | WAREHOUSE | DATABASE, MANAGE GRANTS, MONITOR USAGE...**ON ACCOUNT**



GRANT USAGE | MONITOR | MODIFY | OPERATE **ON WAREHOUSE**



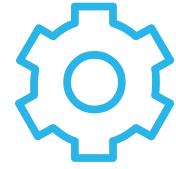
GRANT MODIFY | MONITOR **ON RESOURCE MONITOR**



AVAILABLE PRIVILEGES DEPEND ON THE OBJECT



CREATE ROLE | USER | WAREHOUSE | DATABASE, MANAGE GRANTS, MONITOR USAGE... **ON ACCOUNT**



GRANT USAGE | MONITOR | MODIFY | OPERATE **ON WAREHOUSE**



GRANT MODIFY | MONITOR **ON RESOURCE MONITOR**

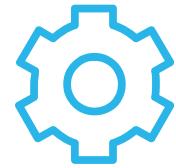


GRANT MODIFY | MONITOR | USAGE | CREATE SCHEMA **ON DATABASE**

AVAILABLE PRIVILEGES DEPEND ON THE OBJECT



CREATE ROLE | USER | WAREHOUSE | DATABASE, MANAGE GRANTS, MONITOR USAGE...**ON ACCOUNT**



GRANT USAGE | MONITOR | MODIFY | OPERATE **ON WAREHOUSE**



GRANT MODIFY | MONITOR **ON RESOURCE MONITOR**



GRANT MODIFY | MONITOR | USAGE | CREATE SCHEMA **ON DATABASE**



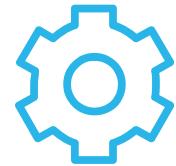
GRANT MODIFY | MONITOR | USAGE | CREATE TABLE | CREATE VIEW...**ON SCHEMA**



AVAILABLE PRIVILEGES DEPEND ON THE OBJECT



CREATE ROLE | USER | WAREHOUSE | DATABASE, MANAGE GRANTS, MONITOR USAGE...**ON ACCOUNT**



GRANT USAGE | MONITOR | MODIFY | OPERATE **ON WAREHOUSE**



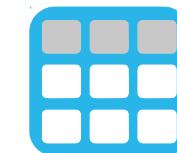
GRANT MODIFY | MONITOR **ON RESOURCE MONITOR**



GRANT MODIFY | MONITOR | USAGE | CREATE SCHEMA **ON DATABASE**



GRANT MODIFY | MONITOR | USAGE | CREATE TABLE | CREATE VIEW...**ON SCHEMA**



GRANT SELECT | INSERT | UPDATE | DELETE | TRUNCATE...**ON TABLE**



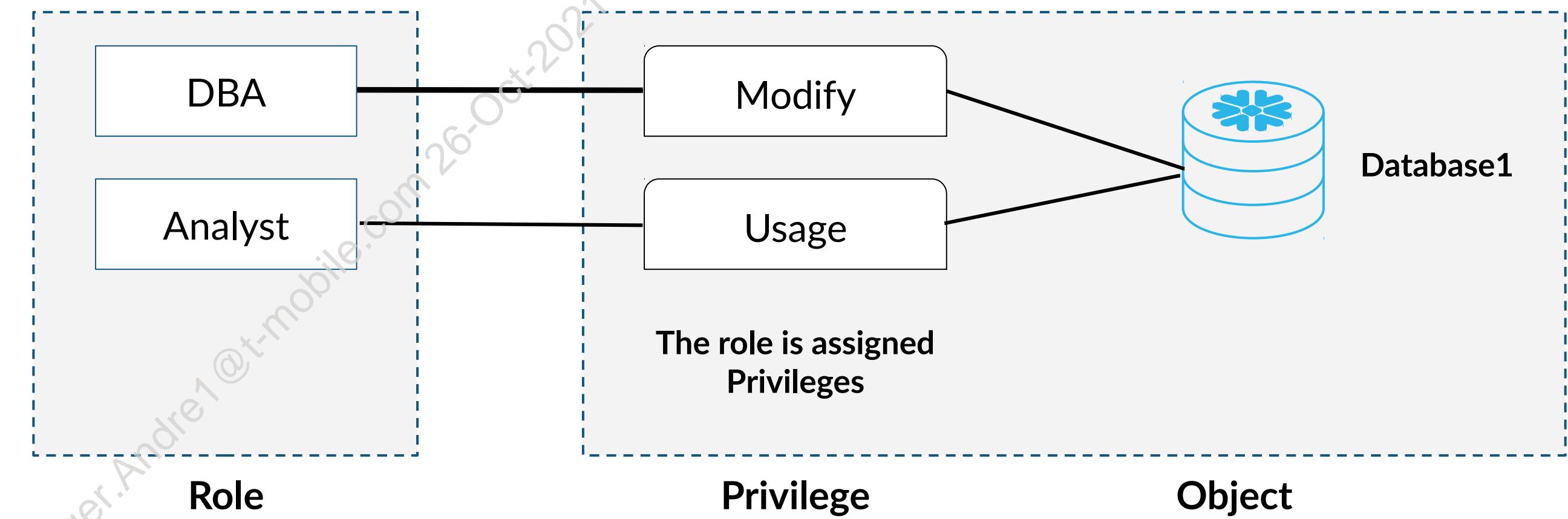
ABOUT PRIVILEGES

- Warehouse privileges USAGE and OPERATE are separate and specific
 - USAGE: use a warehouse to run queries
 - OPERATE: grants ability to suspend, or resume a virtual warehouse
 - MODIFY: grants ability to change the settings or properties of a virtual warehouse
 - For example, you can change the size
- Acting on an object requires USAGE on its logical container(s)
 - Example: to use SELECT privileges on a table, you must have USAGE on the schema and database containing that table
- CREATE, SELECT, INSERT, UPDATE, and DELETE are all separate privileges



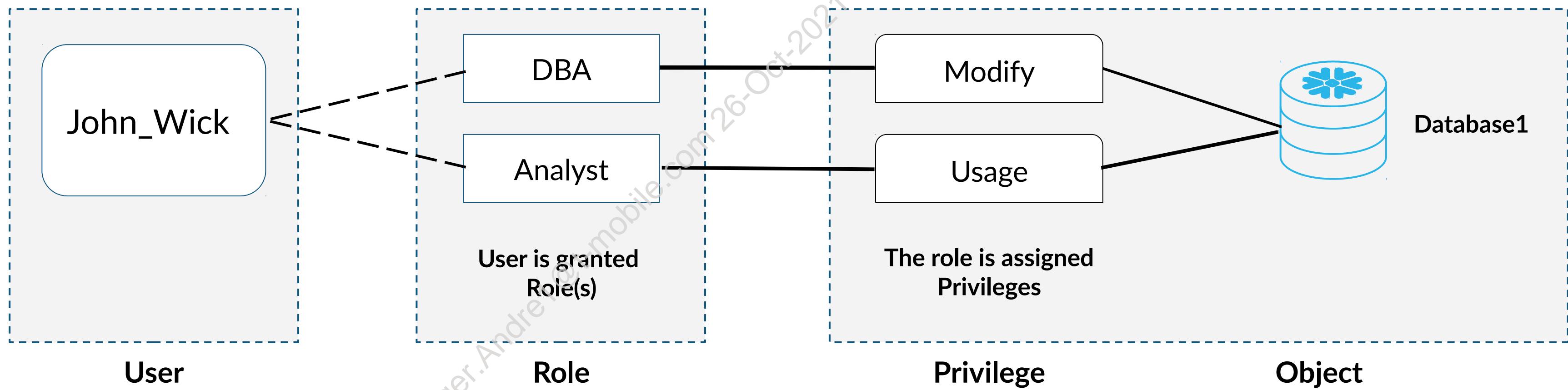
USERS, ROLES, AND GRANTS

- Roles are granted permissions (privileges) to access objects



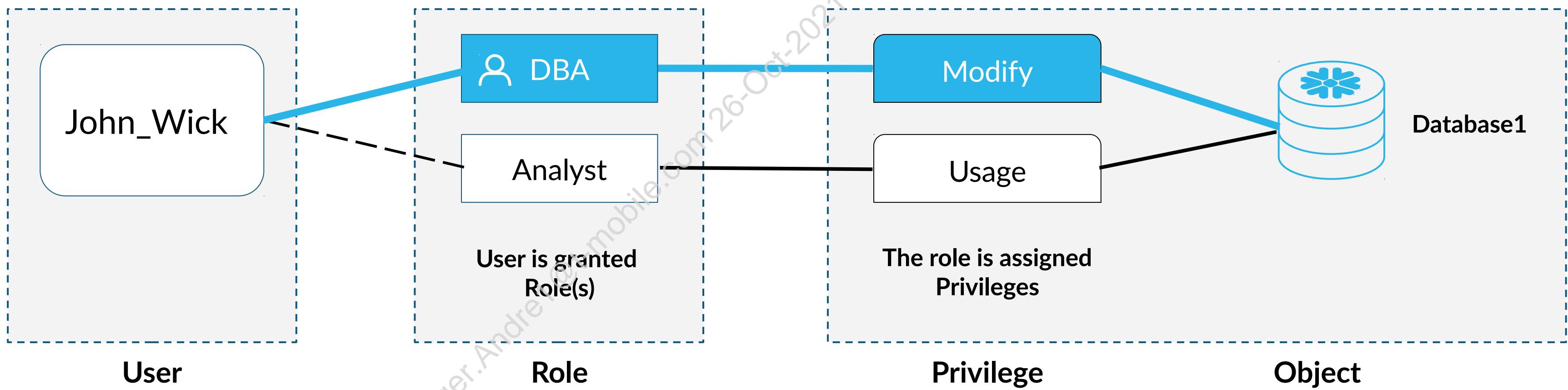
USERS, ROLES, AND GRANTS

- Roles are granted permissions (privileges) to access objects
- Users are granted roles which they can use



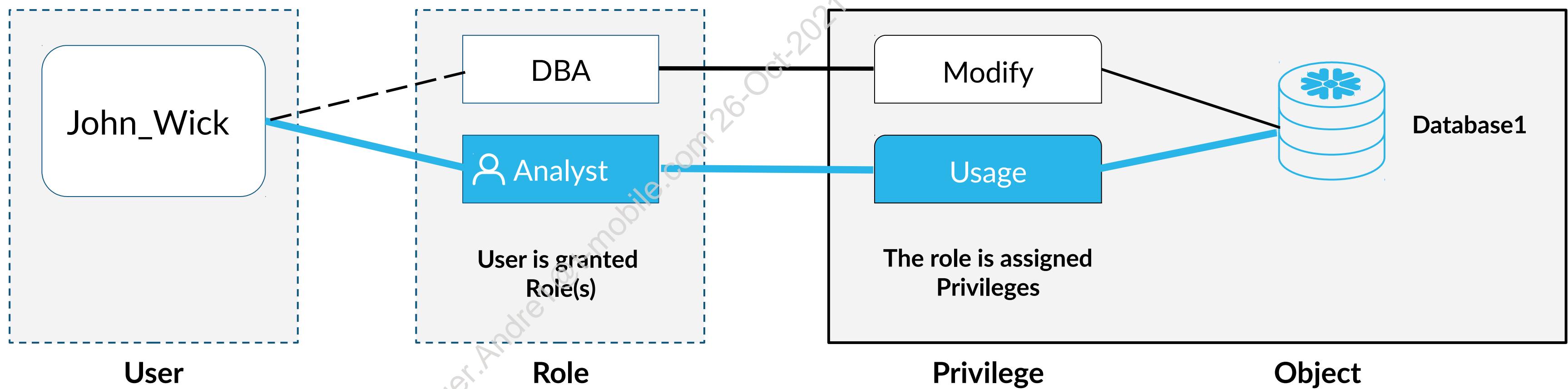
USERS, ROLES, AND GRANTS

- Roles are granted permissions (privileges) to access objects
- Users are granted roles which they can use
- Users USE a role, and get the role's privileges while using the role



USERS, ROLES, AND GRANTS

- Roles are granted permissions (privileges) to access objects
- Users are granted roles which they can use
- Users USE a role, and get the role's privileges while using the role
- Users can be assigned to multiple roles, but only use one (current role) at a time



John has privileges to **USE** the database while using the **Analyst** role

SYSTEM ROLES

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SYSTEM ROLES

ACCOUNTADMIN

SECURITYADMIN

SYSADMIN

USERADMIN

PUBLIC

- ACCOUNTADMIN
 - Has all privileges
- SECURITYADMIN
 - Manages account security plus inherits user and role management from USERADMIN
- USERADMIN
 - Create, monitor, manage users and roles
 - Grant or revoke privileges to users and roles
- SYSADMIN
 - Create warehouses and databases
 - Create other objects
- PUBLIC
 - No privileges
 - Default role for all users, unless otherwise specified



DIVISION OF RESPONSIBILITY

ACCOUNTADMIN

- ACCOUNTADMIN has ALL privileges - either granted or inherited. Grant this role to a limited number of users.

DEFAULT GRANTS:

- CREATE SHARE
- IMPORT SHARE
- MONITOR USAGE

By default, only ACCOUNTADMIN can do these things

ACCOUNTADMIN

SECURITYADMIN

SYSADMIN

USERADMIN



BEST PRACTICES

ACCOUNTADMIN

- Assign to a limited number of users.
- Require MFA on all users with this role.
- Do not make it the default role for any user.
- Never execute scripts using ACCOUNTADMIN.

ACCOUNTADMIN

SECURITYADMIN

SYSADMIN

USERADMIN



DIVISION OF RESPONSIBILITY

SECURITYADMIN

- SECURITYADMIN should manage grants on the accounts, and GRANT them to SYSADMIN (or somewhere in the SYSADMIN hierarchy).
- SECURITYADMIN can optionally be used to conduct any USERADMIN functions.

DEFAULT GRANTS:

- Inherits all user and role management from USERADMIN.
- Manage network policies.
- MANAGE GRANTS privileges.

ACCOUNTADMIN

SECURITYADMIN

SYSADMIN

USERADMIN



DIVISION OF RESPONSIBILITY

USERADMIN

- USERADMIN should create all users and roles, and GRANT them to SYSADMIN (or somewhere in the SYSADMIN hierarchy).

Best practice:

- USERADMIN should GRANT any object-related roles to SYSADMIN.

DEFAULT GRANTS:

- CREATE USER ON ACCOUNT.
- CREATE ROLE ON ACCOUNT.

ACCOUNTADMIN

SECURITYADMIN

SYSADMIN

USERADMIN



DIVISION OF RESPONSIBILITY

SYSADMIN

- SYSADMIN should grant PRIVILEGES to ROLES.

DEFAULT GRANTS:

- CREATE DATABASE on ACCOUNT.
- CREATE WAREHOUSE on ACCOUNT.

ACCOUNTADMIN

SECURITYADMIN

SYSADMIN

USERADMIN



DIVISION OF RESPONSIBILITY

PUBLIC

- PUBLIC can't do anything but log in. It's the default role for new users, unless changed.
- PUBLIC is inherited by every role and can be used as a default role to provide access to an object for every user.

DEFAULT GRANTS:

- Zero
- Zip
- Zilch
- Nada

PUBLIC

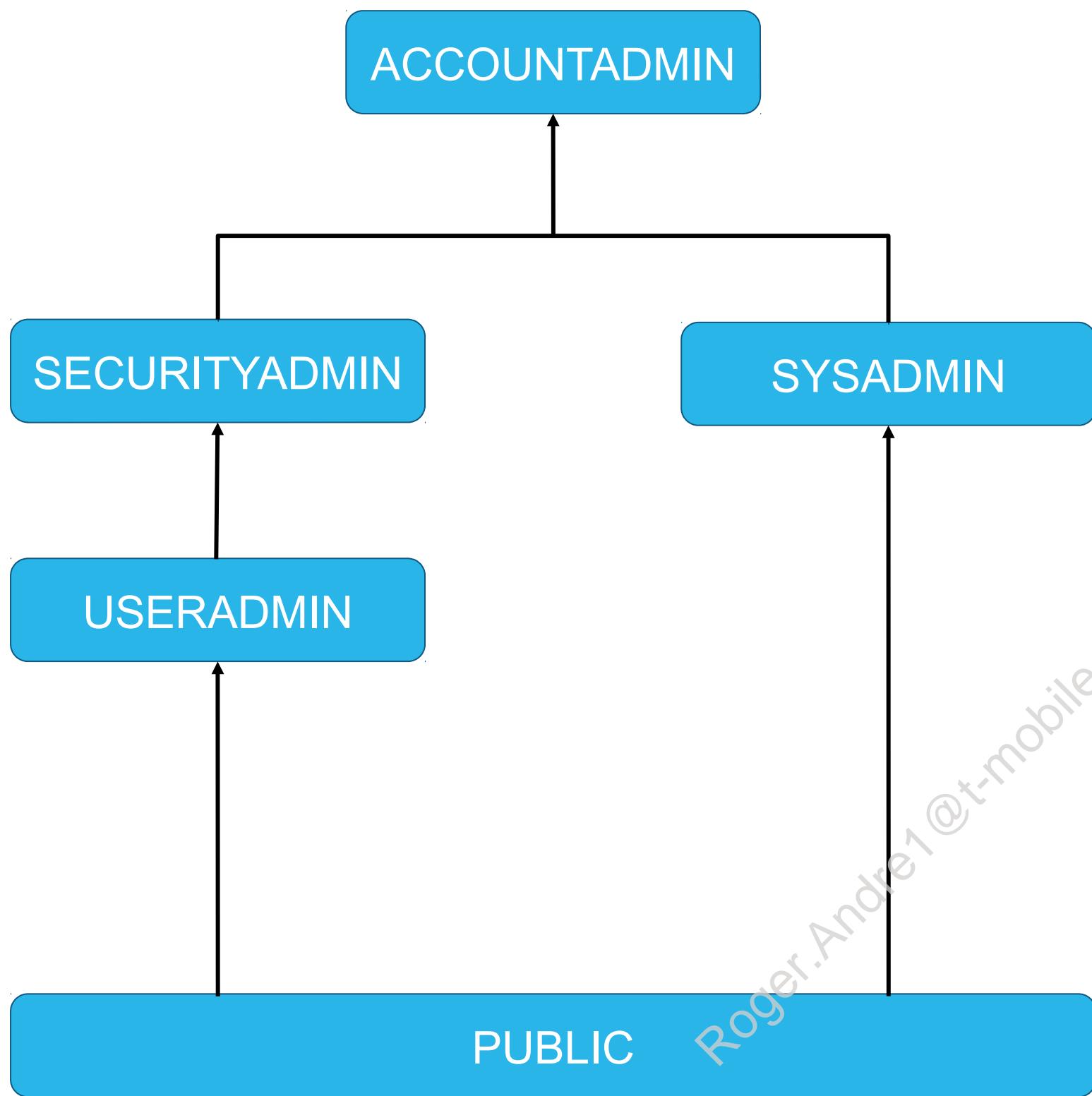


CUSTOM ROLES AND INHERITANCE

Roger.Andre1@t-mobile.com 26-Oct-2021 © Snowflake 2020-do-not-copy



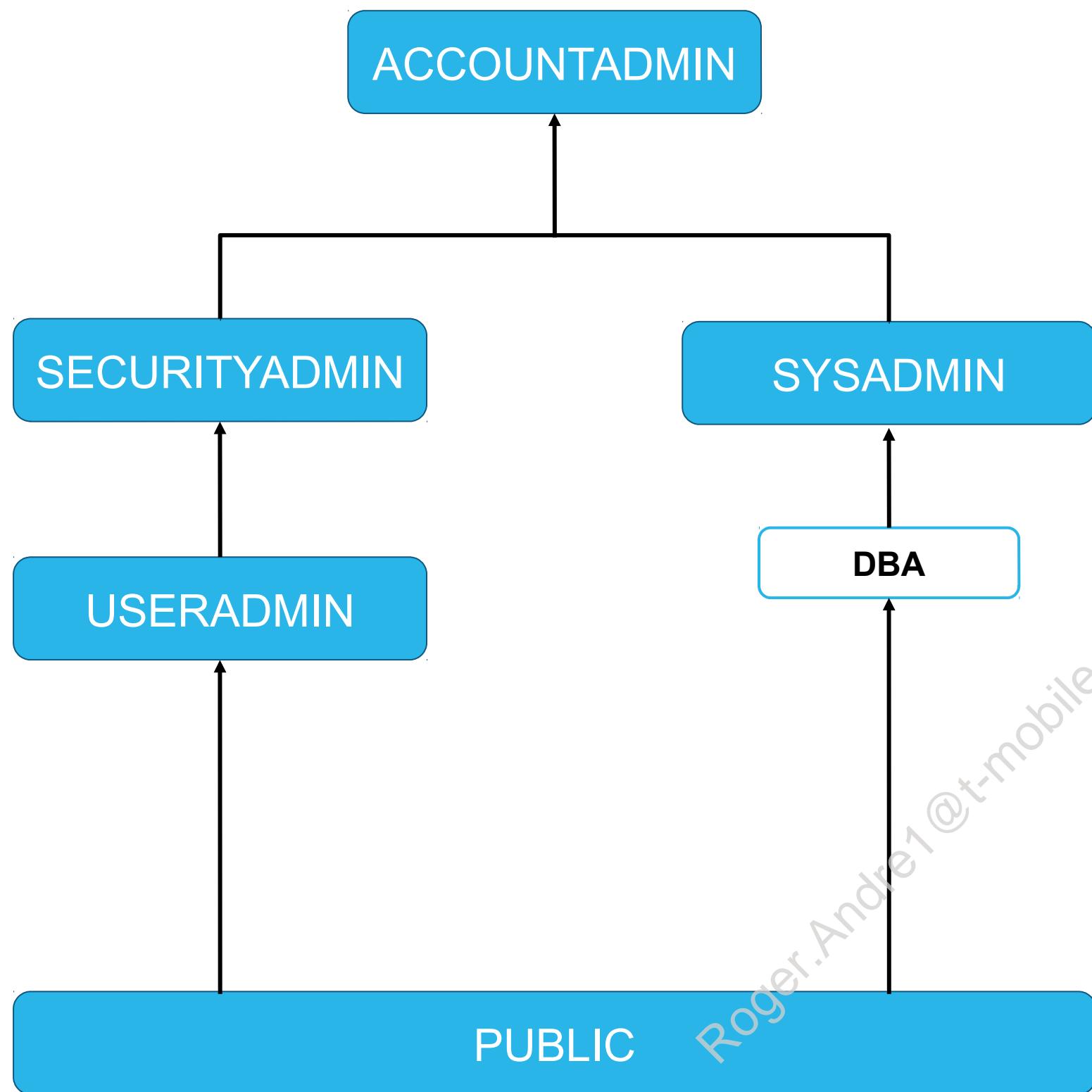
ROLE INHERITANCE



- A role inherits all the privileges of its underlying roles (those “lower” in the hierarchy)
- ACCOUNTADMIN inherits privileges from SECURITYADMIN, USERADMIN, SYSADMIN, and PUBLIC
- SECURITYADMIN inherits privileges from USERADMIN and PUBLIC
- USERADMIN and SYSADMIN inherit privileges from PUBLIC
- PUBLIC inherits nothing.
 - All roles inherit from PUBLIC.



ROLE HIERARCHY



- Roles fit into the hierarchy based on which role(s) they are granted to
- Custom roles also inherit privileges from the roles “beneath” them
- Here:
 - Role DBA has been granted to role SYSADMIN
 - Role SYSADMIN inherits privileges from role DBA
 - Every role inherits privileges from role PUBLIC



CREATING CUSTOM ROLES

```
USE ROLE useradmin;
```

```
CREATE ROLE dba;
```

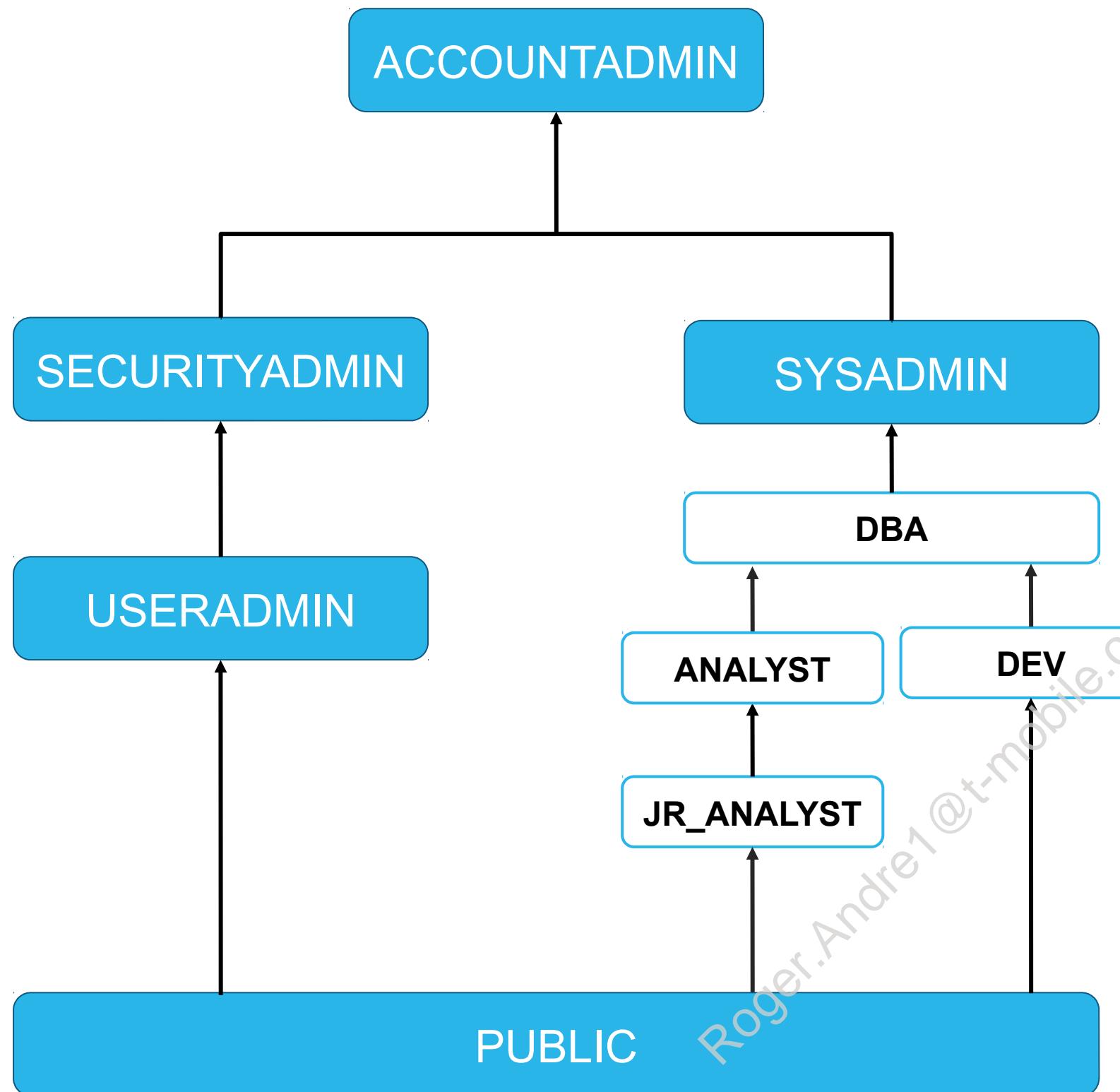
```
GRANT ROLE dba TO ROLE sysadmin;
```

```
GRANT ROLE dba TO USER jenna;
```

```
GRANT ROLE dba TO USER tom;
```



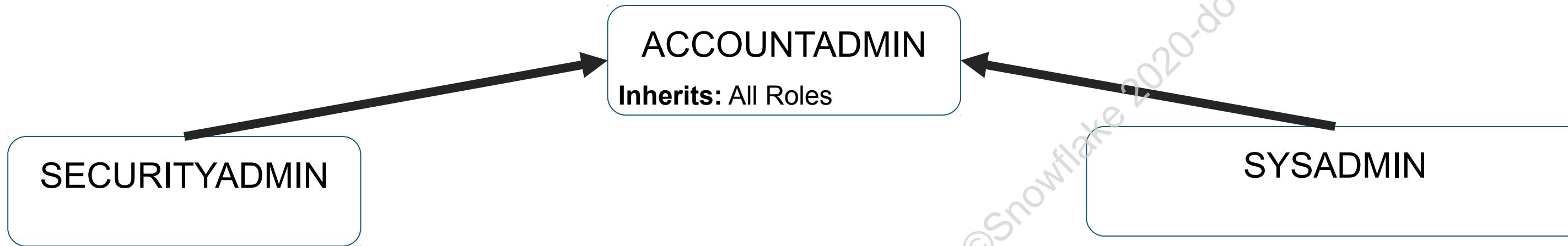
ROLE HIERARCHY



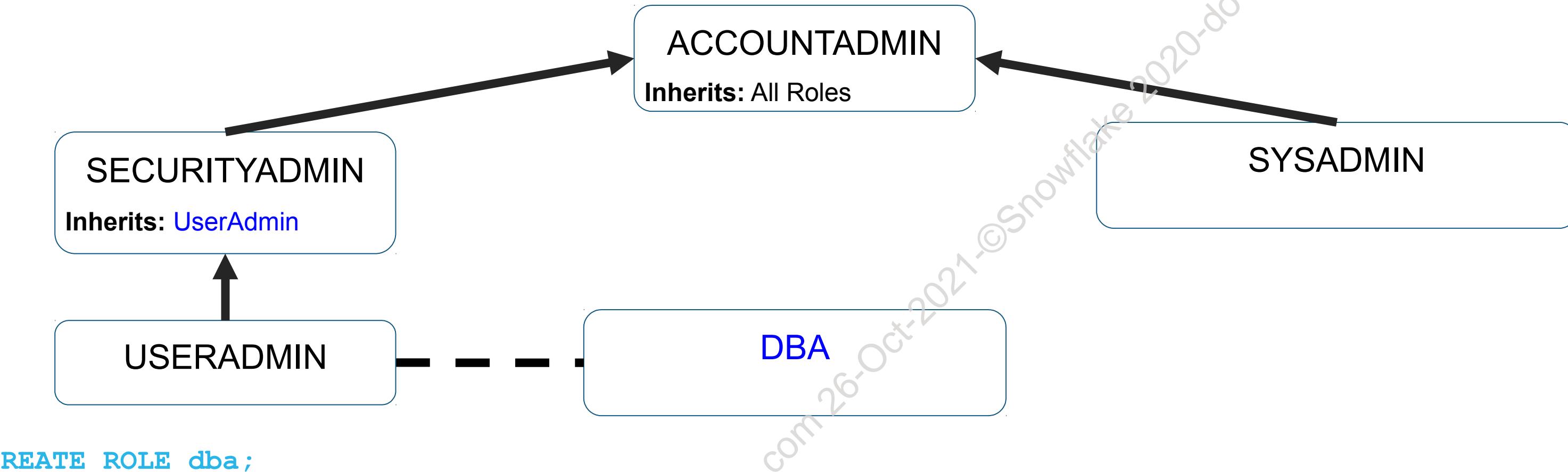
- Grants:
 - Role DBA has been granted to Role SYSADMIN
 - Role ANALYST has been granted to role DBA
 - Role DEV has been granted to role DBA
 - Role JR_ANALYST has been granted to role ANALYST
- Inheritance:
 - SYSADMIN inherits from DBA and roles below it
 - DBA inherits from DEV, ANALYST, and roles below them
 - ANALYST inherits from JR_ANALYST
 - Every role inherits from PUBLIC



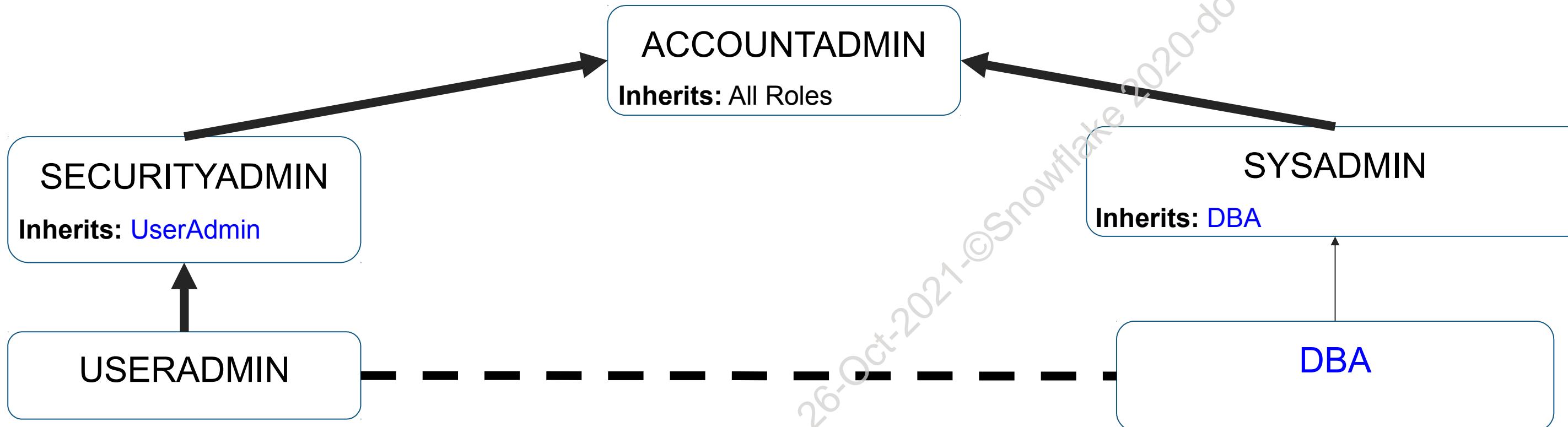
ROLE HIERARCHY VISUALIZATION



ROLE HIERARCHY VISUALIZATION



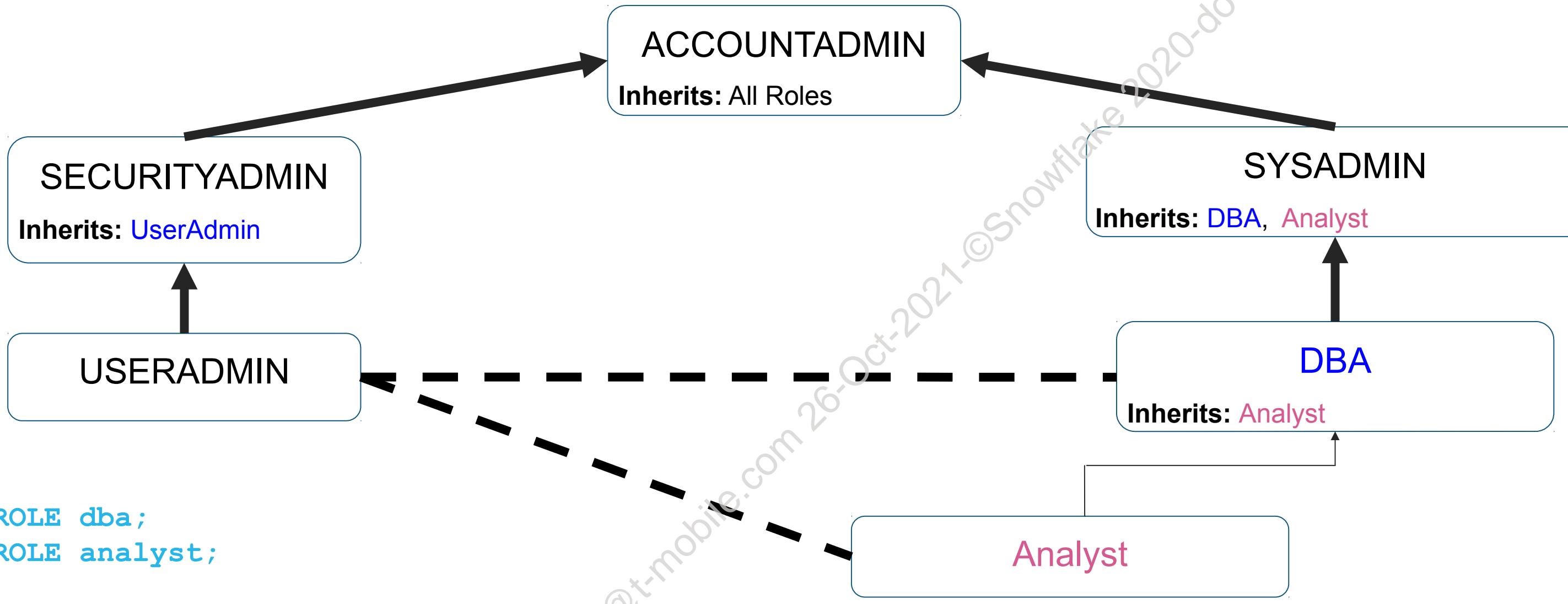
ROLE HIERARCHY VISUALIZATION



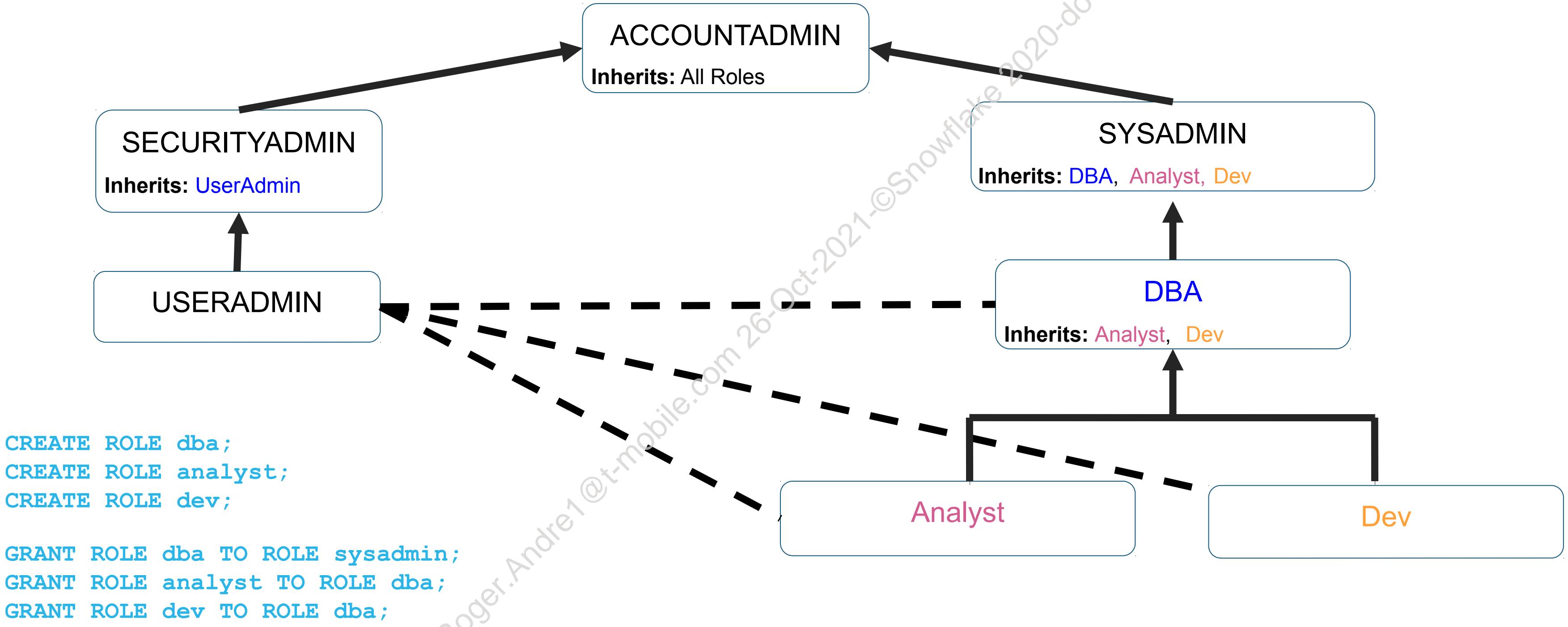
```
GRANT ROLE dba TO ROLE sysadmin;
```



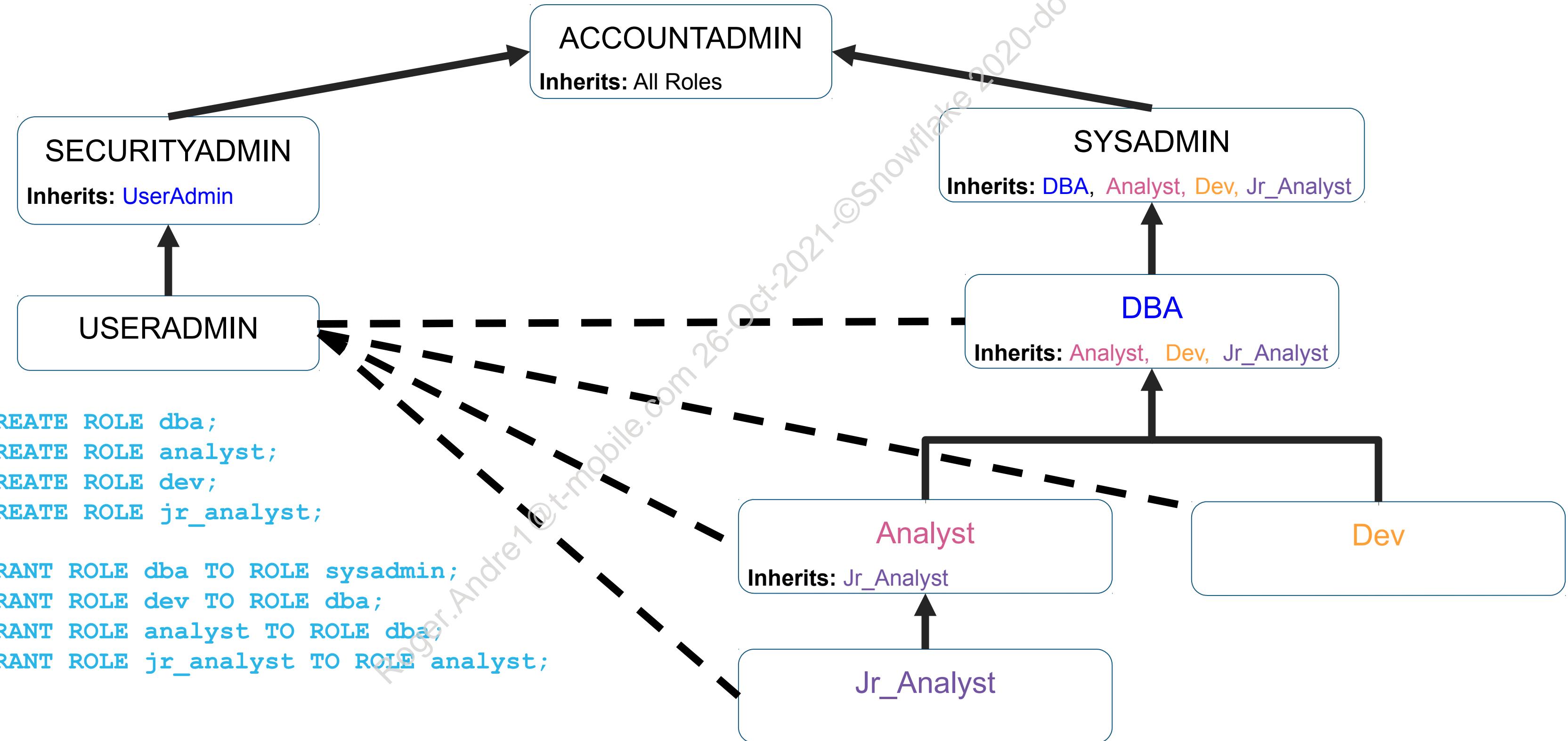
ROLE HIERARCHY VISUALIZATION



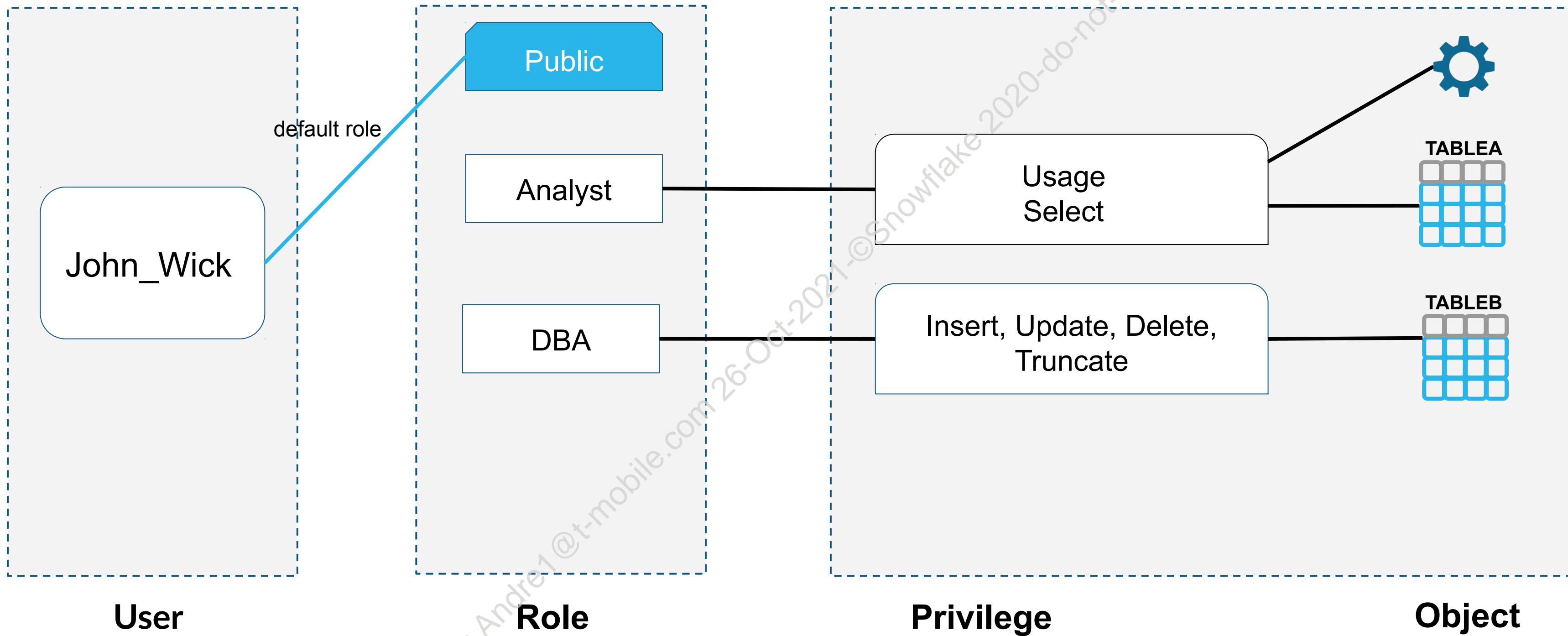
ROLE HIERARCHY VISUALIZATION



ROLE HIERARCHY VISUALIZATION



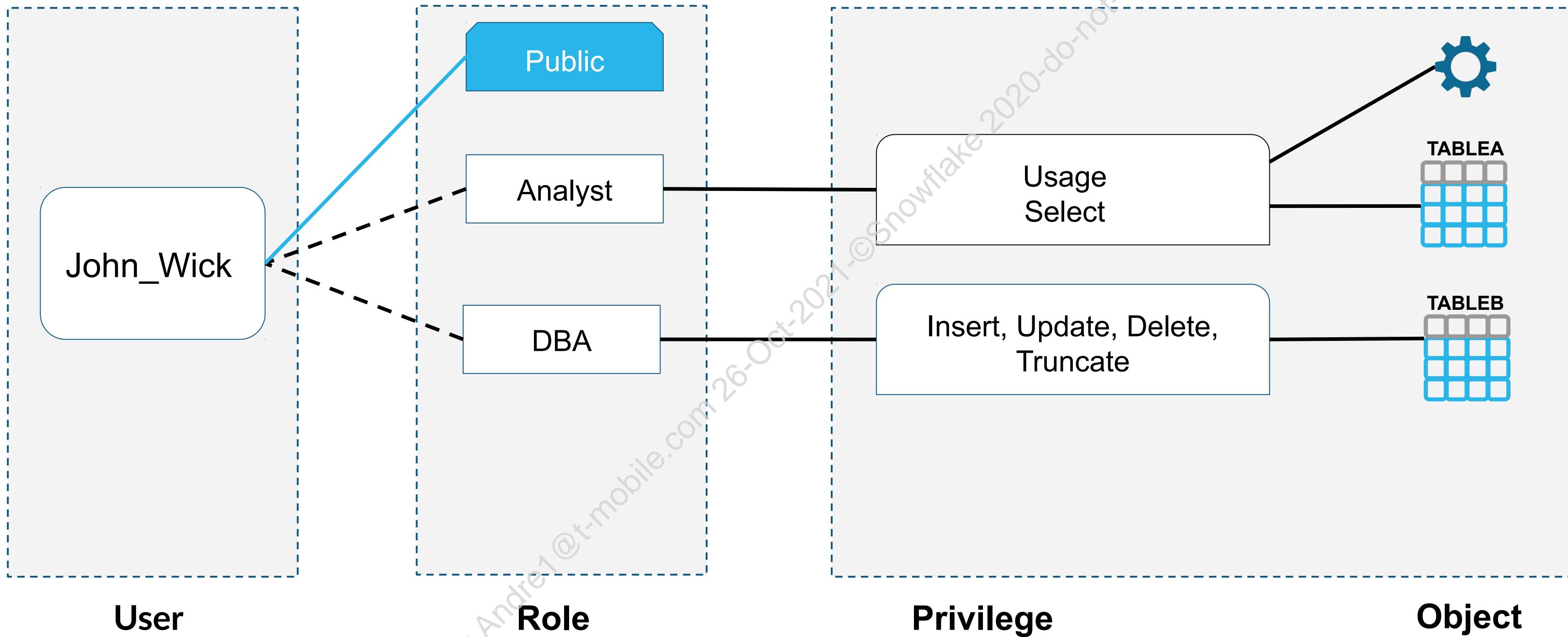
ROLES AND INHERITANCE



Upon connecting to Snowflake, a user takes on their default role.

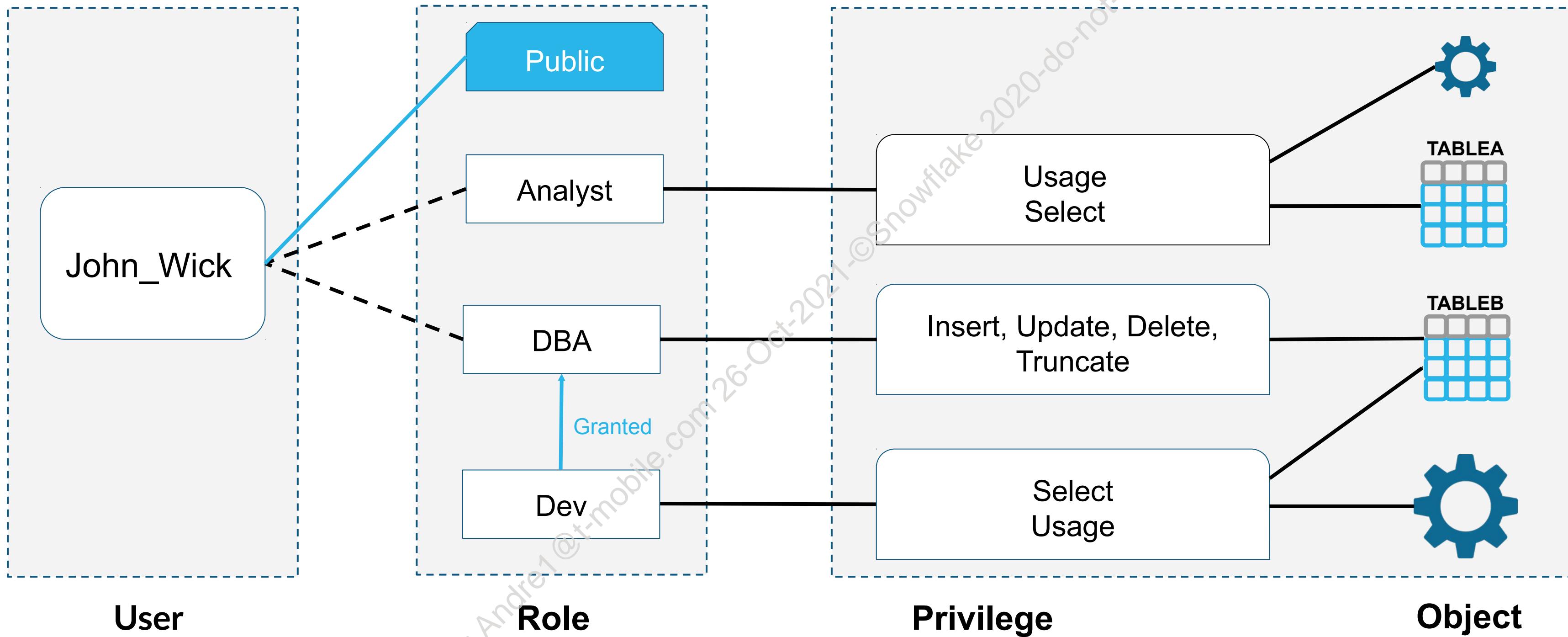


ROLES AND INHERITANCE



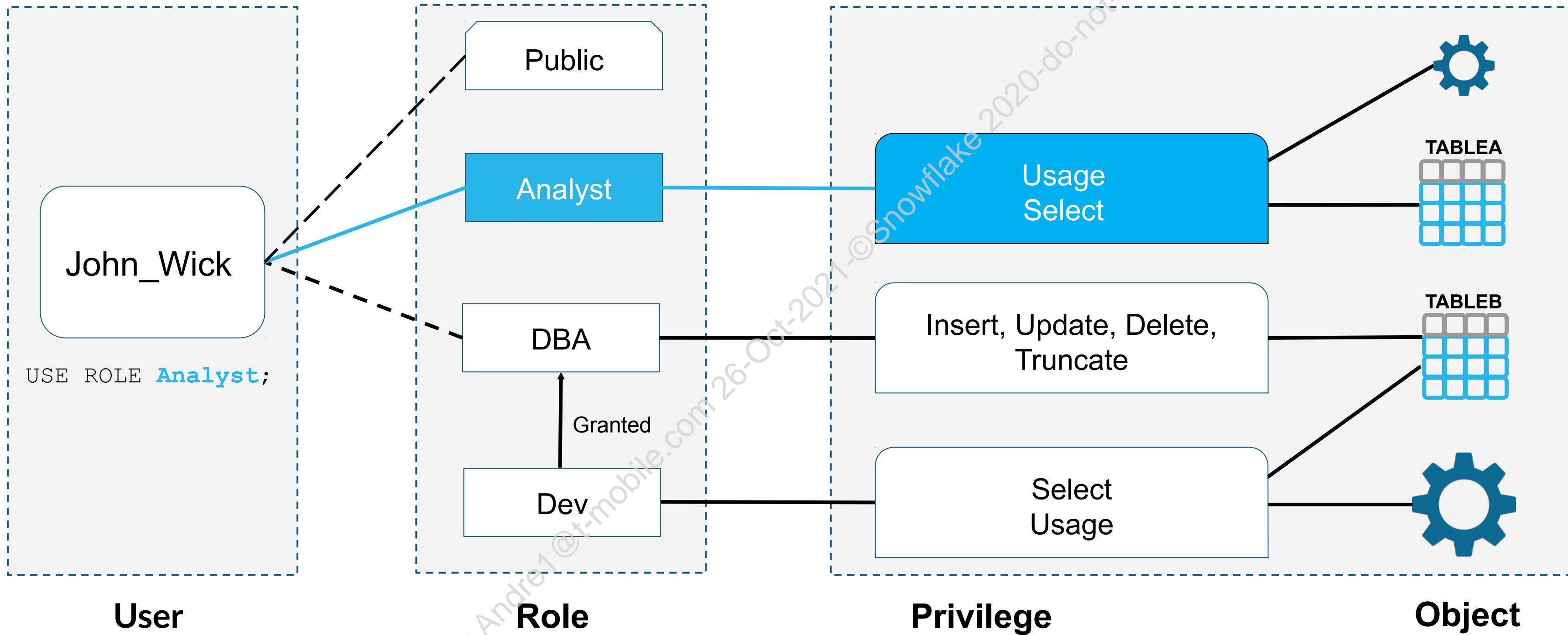
User John_Wick has been granted the roles Analyst and DBA.

ROLES AND INHERITANCE



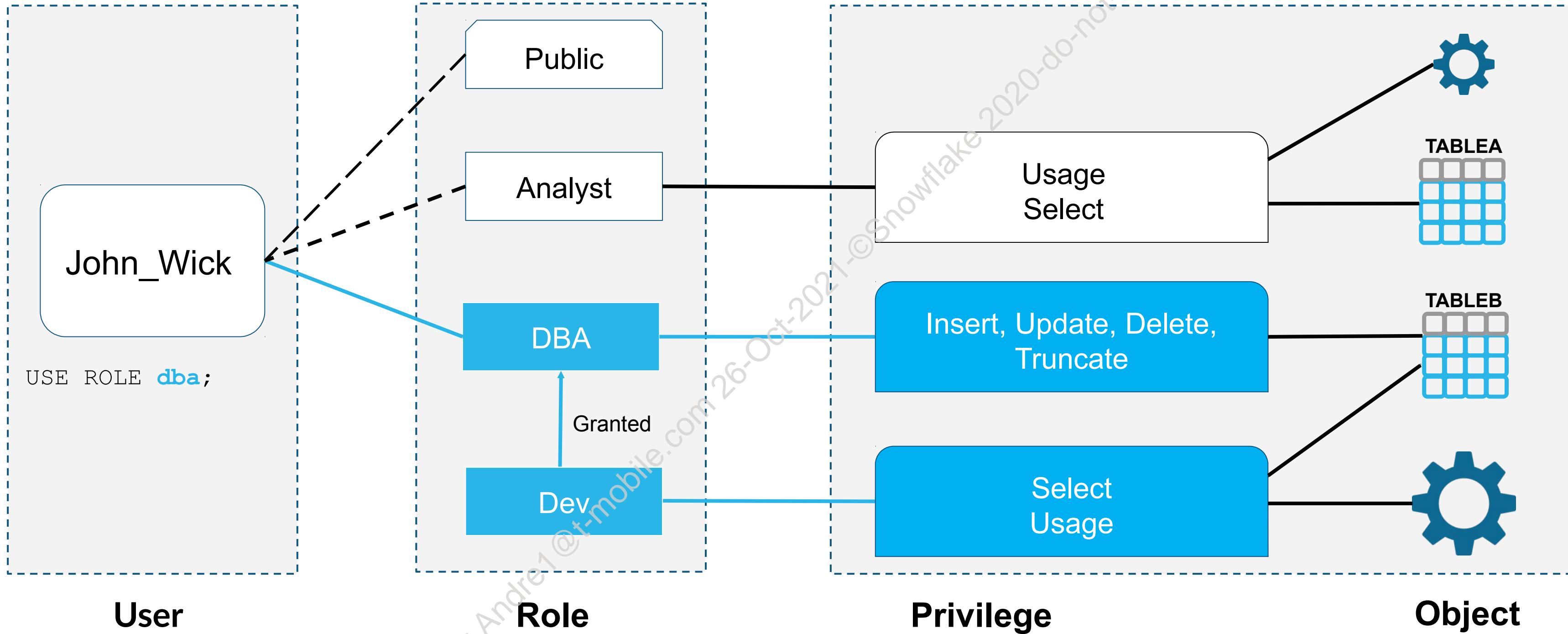
The role **Dev** has been **granted** to the role **DBA**.

ROLES AND INHERITANCE



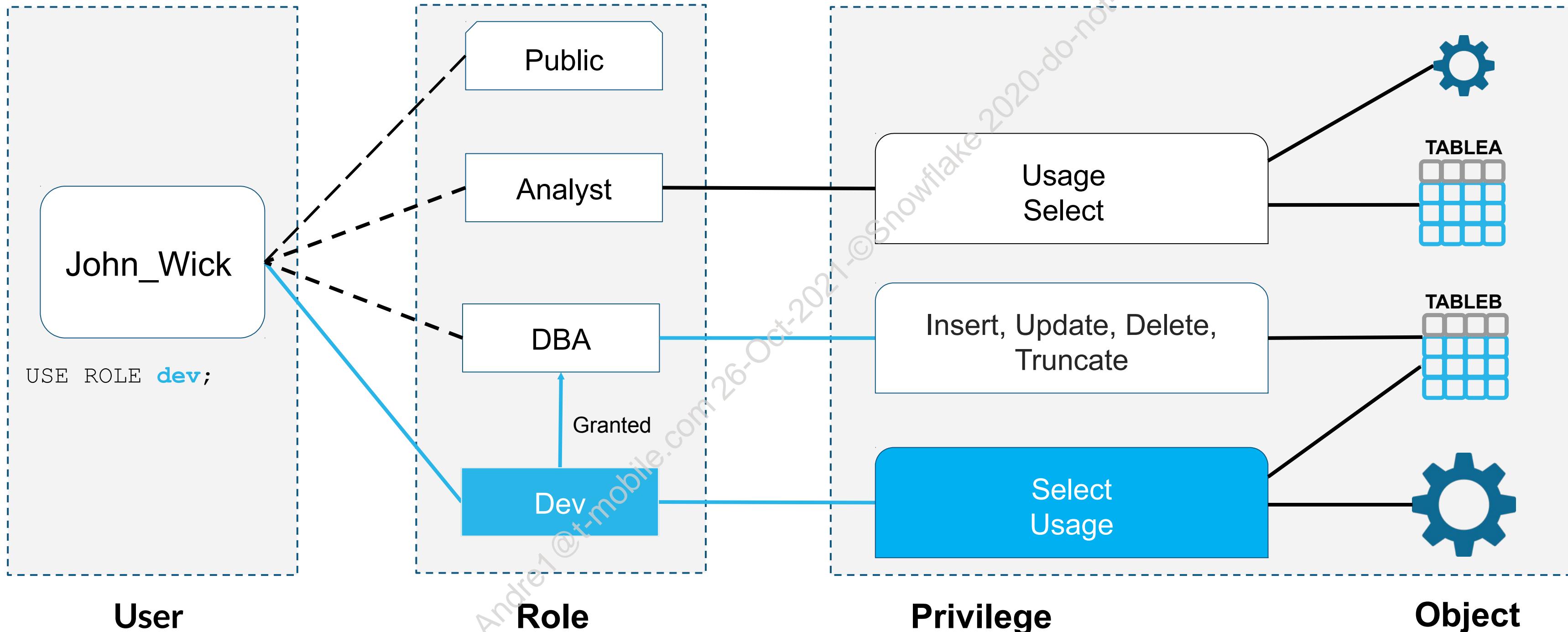
When using the **Analyst** role, John can **SELECT** from **TABLEA** and **USE** a **warehouse**.

ROLES AND INHERITANCE



With the **DBA** role, he can **INSERT**, **UPDATE**, **DELETE**, and **TRUNCATE TABLEB**. He also inherits the right to **SELECT** from **TABLEB**, and **USE** a **warehouse**.

ROLES AND INHERITANCE



He can also **USE** the Dev role, since it was granted to a role he has the right to use. As Dev, He can **SELECT** from **TABLEB** and **USE** a **warehouse**.

OWNERSHIP

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



OWNERSHIP

- When you create an **object**, your **current role** becomes the **owning role**
- Every role or object is owned by a single role.
 - All users in that role share ownership, when they are using that role.
- The **owning role** can do anything with the object.
- The **owning role** can grant privileges to itself or other roles.
- Ownership can be transferred by the owning role.

```
USE ROLE SYSADMIN; -- Owner of the object
```

```
GRANT OWNERSHIP ON WAREHOUSE dev_ws TO ROLE dba
```



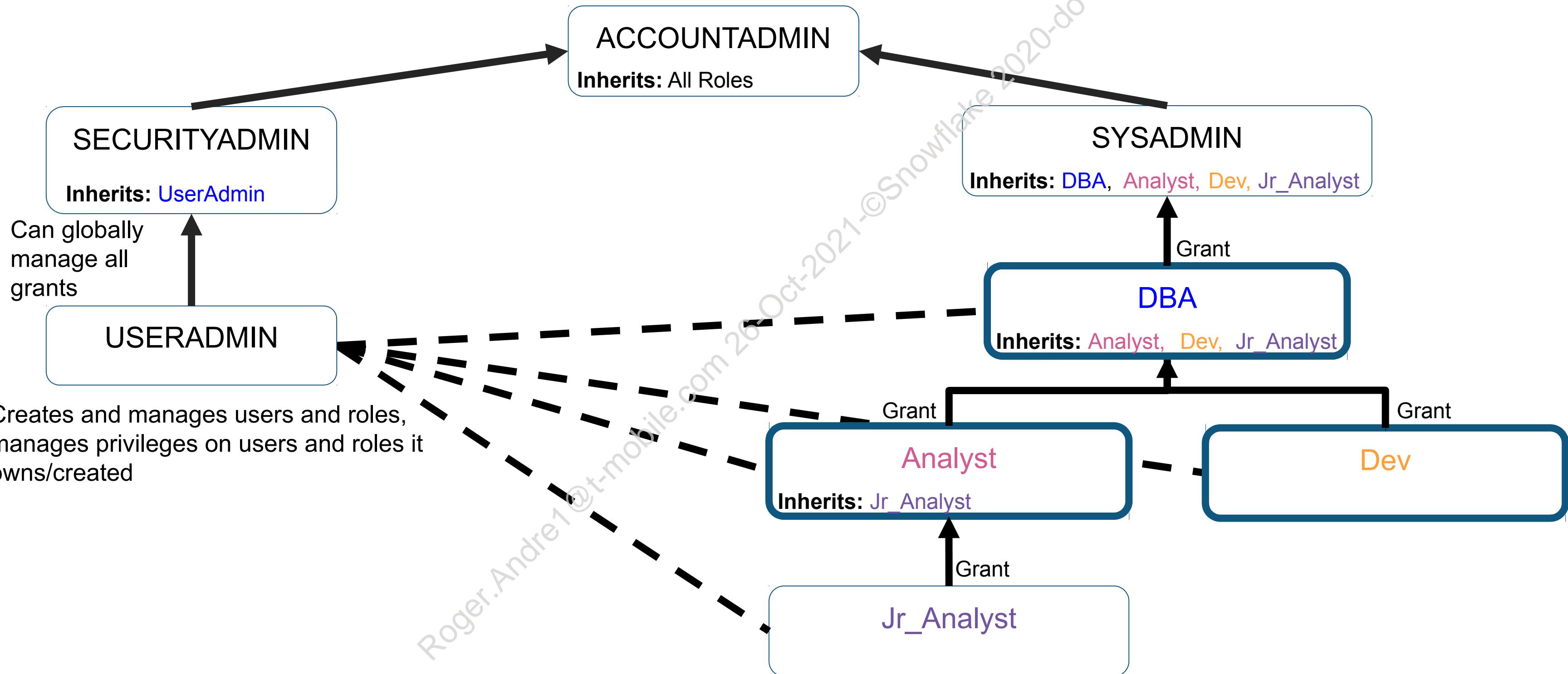
OWNERSHIP

- **Warning:** Object ownership may cause confusion
 - Be aware of which role you're using when you create an object
 - When you change roles, your object may no longer be accessible (the object belongs to your role when you created it, it does not belong to you)
 - If an object goes “missing”, check with the role owner
- Best Practice: USERADMIN creates all users; this is a change from earlier
- Example:

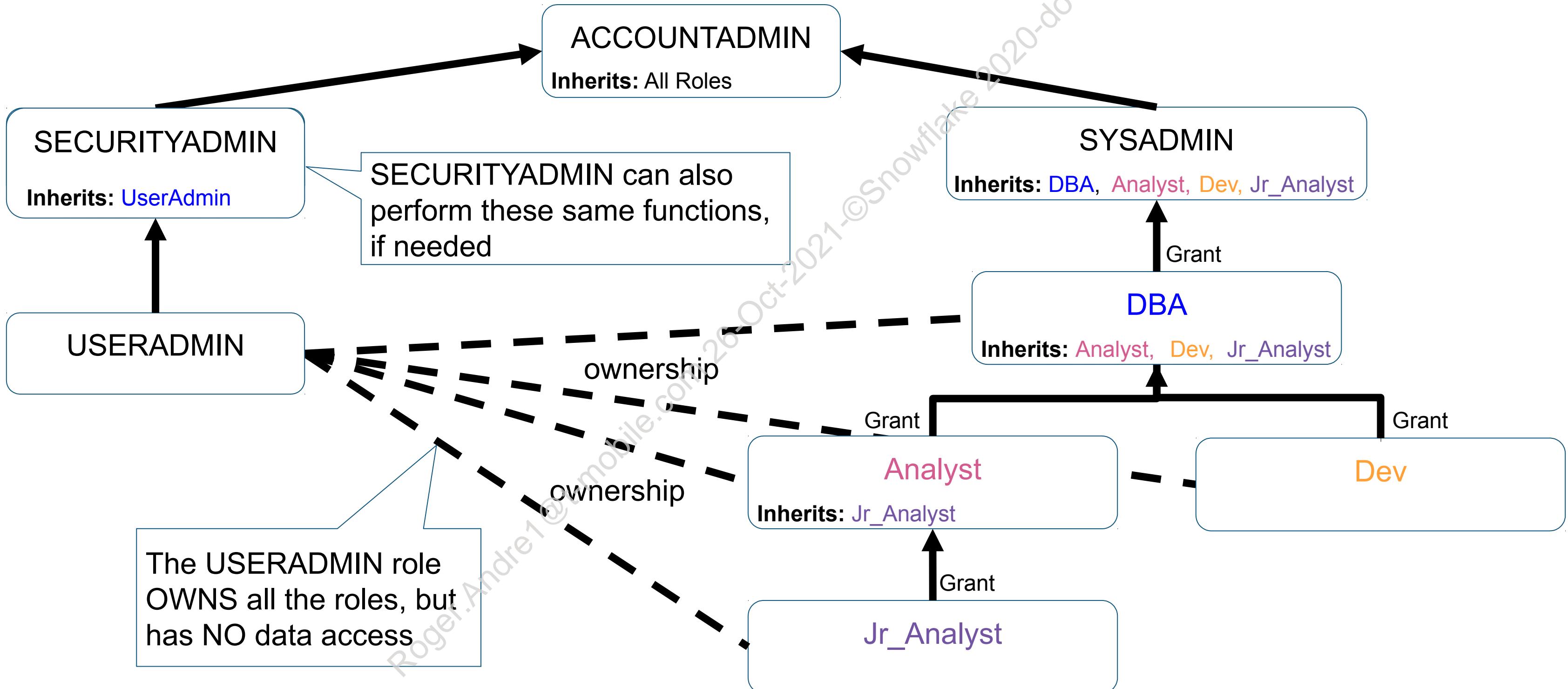
```
GRANT CREATE DATABASE ON ACCOUNT TO ROLE dba;  
USE ROLE dba;  
CREATE DATABASE my_db1;  
CREATE SCHEMA my_db1.myschema; -- privilege acquired through ownership
```



ROLE OWNERSHIP VERSUS GRANT



ROLE OWNERSHIP VERSUS GRANT



OWNERSHIP AND GRANTS

- Granted privileges are different from ownership
- GRANTs give a role the right to do something with an object or to use a role
- The OWNER has the right to GRANT privileges on the objects they own
- The OWNER also has all rights to the objects they own



CONFIGURE AND MANAGE ACCESS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



CREATE USERS AND ROLES

```
USE ROLE useradmin;
```

```
CREATE USER johnsmith
  PASSWORD = 'RoseBud1'
  default_role = developer
  must_change_password = true;
```

```
DESC USER johnsmith;
```

```
CREATE ROLE dba;
```

```
GRANT ROLE dba TO USER johnsmith;
```

- Users with the USERADMIN role or higher can manage users and roles using SQL on the web UI.
- Best practice for creating a user is to force a password change on first login.
- In this example, since the developer role has not been granted to johnsmith, if the johnsmith will not be able to login to the WebUI until granted to the developer role.
- Use USERADMIN to create users and roles, optionally SECURITYADMIN can also perform this function



MANAGE USERS

```
ALTER USER johnsmith  
  SET PASSWORD = 'Rosebud2'  
  must_change_password = true;
```

```
ALTER USER johnsmith  
  RESET PASSWORD;
```

```
ALTER USER johnsmith  
  SET disabled = true;
```

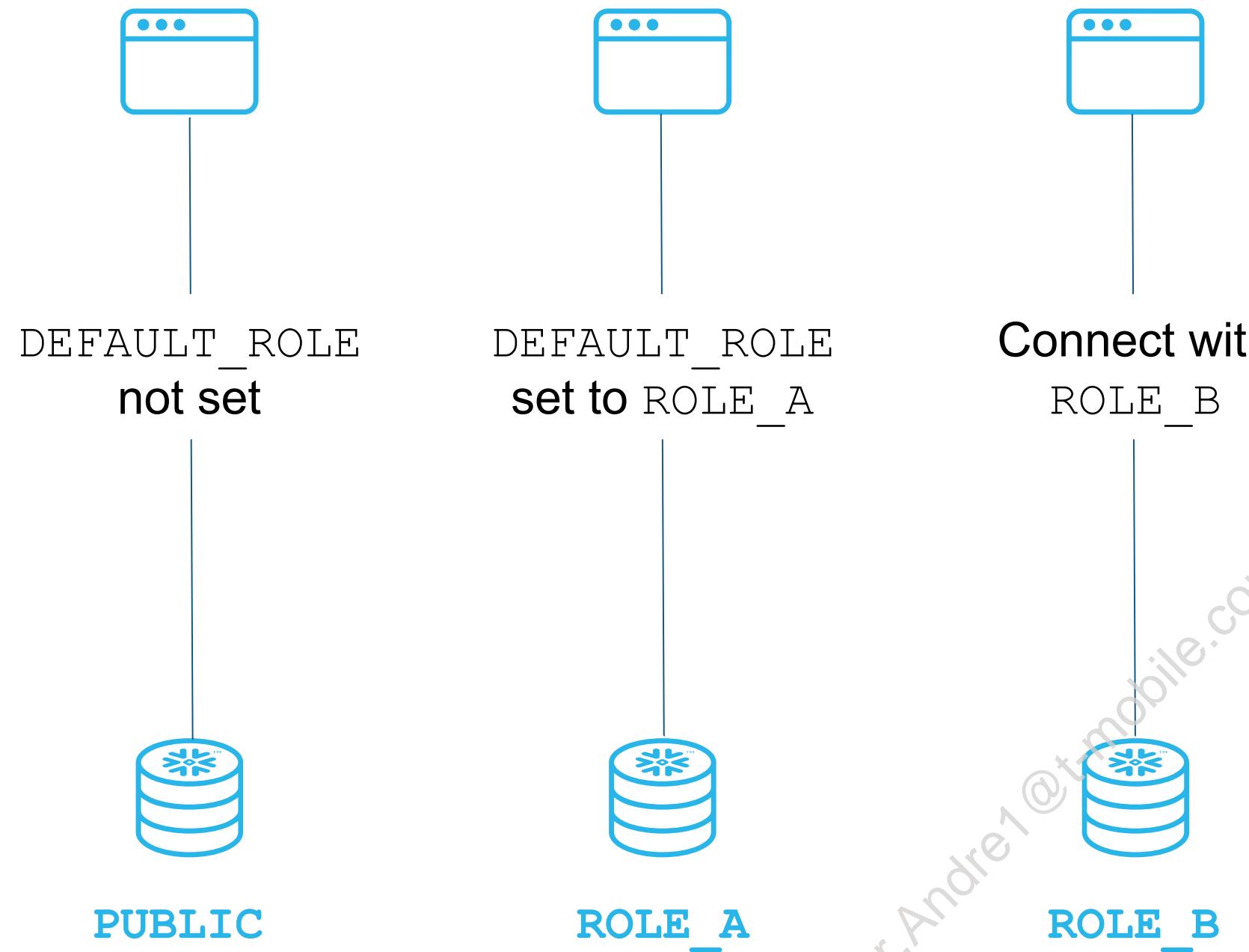
```
ALTER USER johnsmith  
  SET MINS_TO_UNLOCK = 0;
```

```
ALTER USER johnsmith  
  SET PASSWORD = 'Citizen1';
```

- Use ALTER USER to change the password for another user
- After 5 login failures, user is locked out for 15 minutes
- RESET PASSWORD generates a URL to share with the user
 - Old password is valid until changed by user
 - URL expires in 4 hours
- Disabling a user terminates user sessions/locks them out immediately



ASSIGN A DEFAULT ROLE



- Default role is **PUBLIC**
- Set **DEFAULT_ROLE** for user to change the default
`ALTER USER name SET DEFAULT_ROLE role`
- Set the role in the connection string from the client (JDBC, ODBC, SnowSQL...)
 - Role set in connection string overrides configured default role



VIEW CURRENT PRIVILEGES

- View the current roles granted to a user:

```
SHOW GRANTS TO USER <user_name>;
```

- View the current set of privileges granted to a role:

```
SHOW GRANTS TO ROLE <role_name>;
```

- View which users a role has been granted to:

```
SHOW GRANTS OF ROLE <role_name>;
```



SHOW GRANTS ON OBJECTS

- Lists all the account-level (i.e. global) privileges that have been granted to roles:

SHOW GRANTS ON ACCOUNT;

- Lists all privileges that have been granted on an object:

SHOW GRANTS ON <object_type> <object_name>;



GRANT PRIVILEGES

```
GRANT ROLE warehouse_manager TO USER kelly;
```

```
GRANT ROLE dba TO USER allison;
```

```
GRANT ROLE analyst TO USER marie;
```

```
GRANT OPERATE ON WAREHOUSE wh1 TO ROLE warehouse_manager;
```

```
GRANT USAGE ON WAREHOUSE wh1 TO ROLE dba, analyst;
```

```
GRANT CREATE SCHEMA ON DATABASE db1 TO ROLE dba;
```

```
GRANT SELECT ON ALL TABLES IN SCHEMA schema1 TO ROLE analyst;
```



FUTURE GRANTS

- Grants, by default, apply only to existing objects

```
GRANT SELECT ON ALL TABLES IN SCHEMA mydb.myschema TO ROLE role1;
```

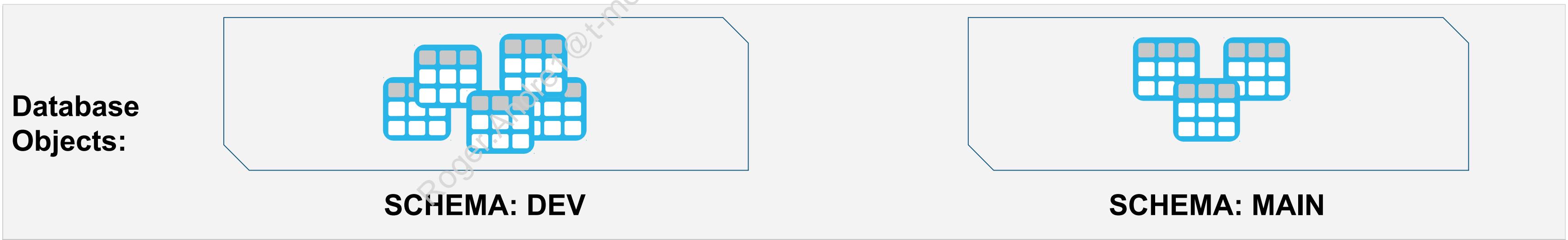
- Use **ON FUTURE** to grant privileges to objects that will be created in the future
 - See documentation for details on future grants

```
GRANT SELECT ON FUTURE TABLES IN SCHEMA mydb.myschema TO ROLE role1;
```

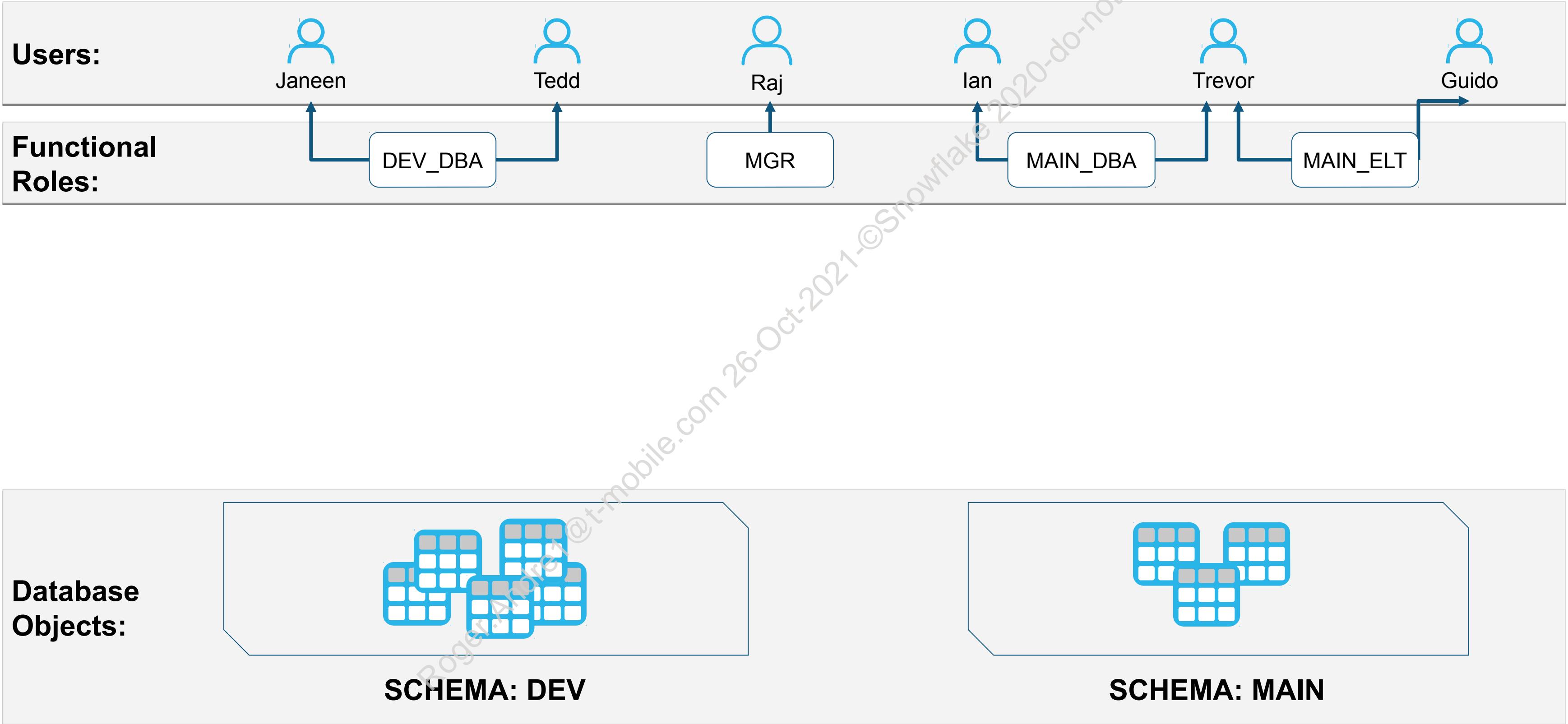
NOTE: To apply FUTURE GRANTS to a SCHEMA requires a ROLE with MANAGED GRANT access or the SCHEMA having MANAGED ACCESS.



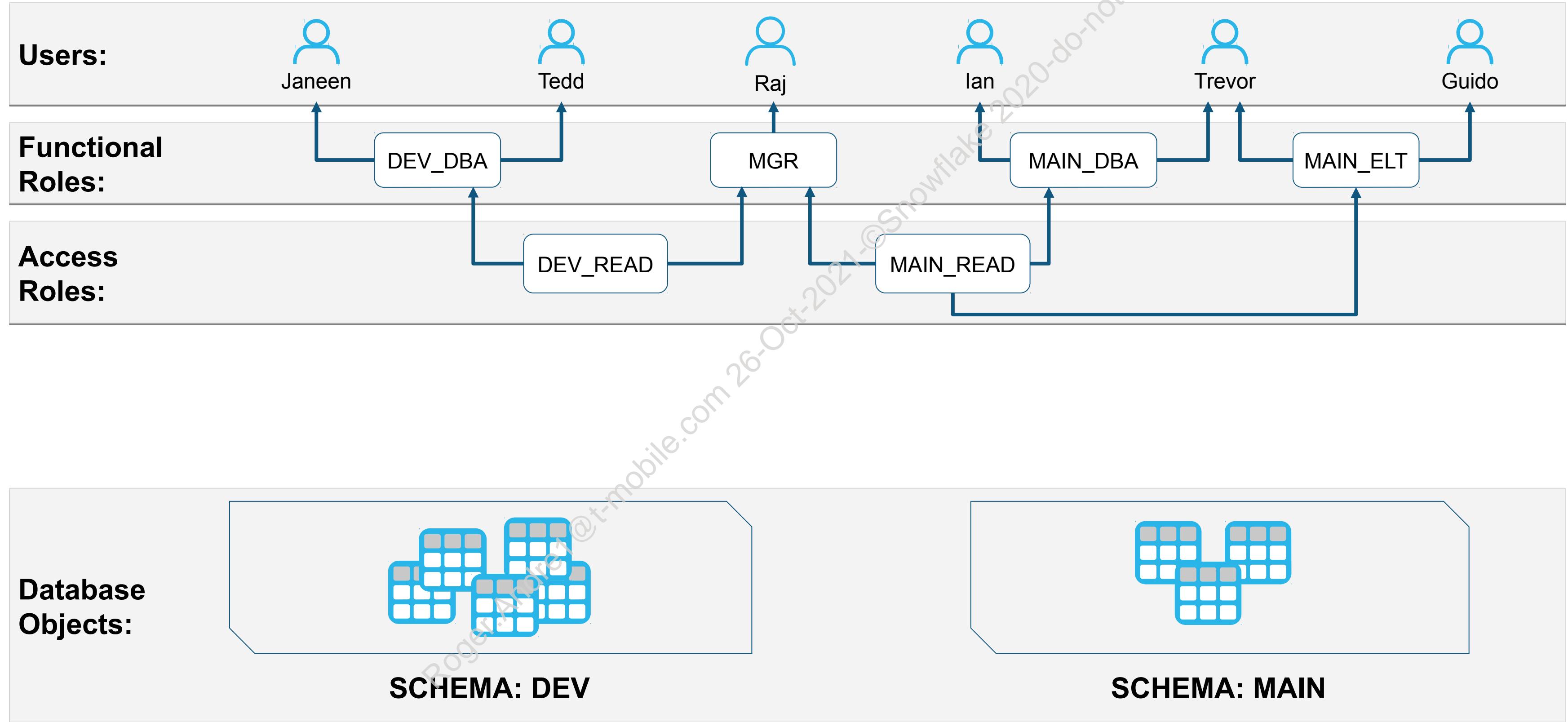
LAYERED ACCESS CONTROL



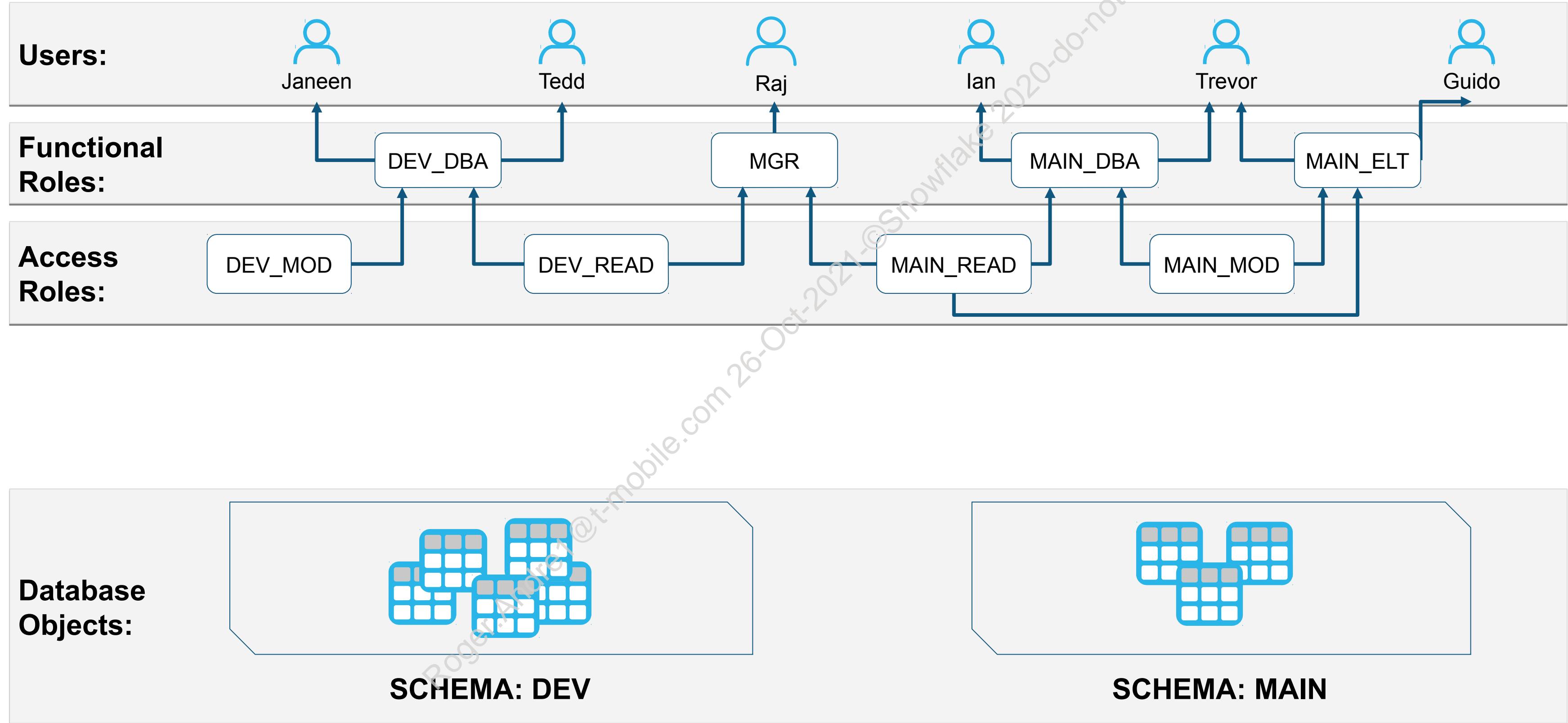
LAYERED ACCESS CONTROL



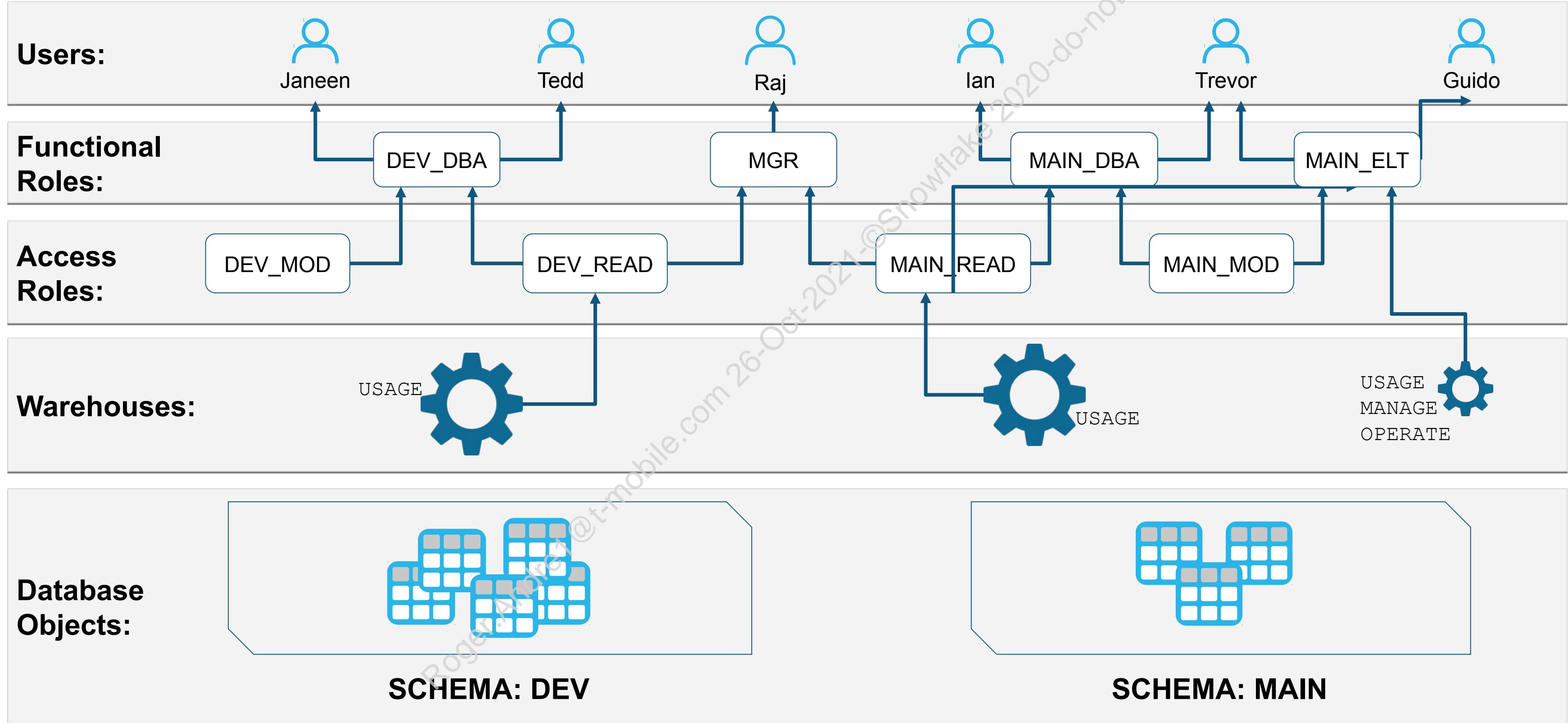
LAYERED ACCESS CONTROL



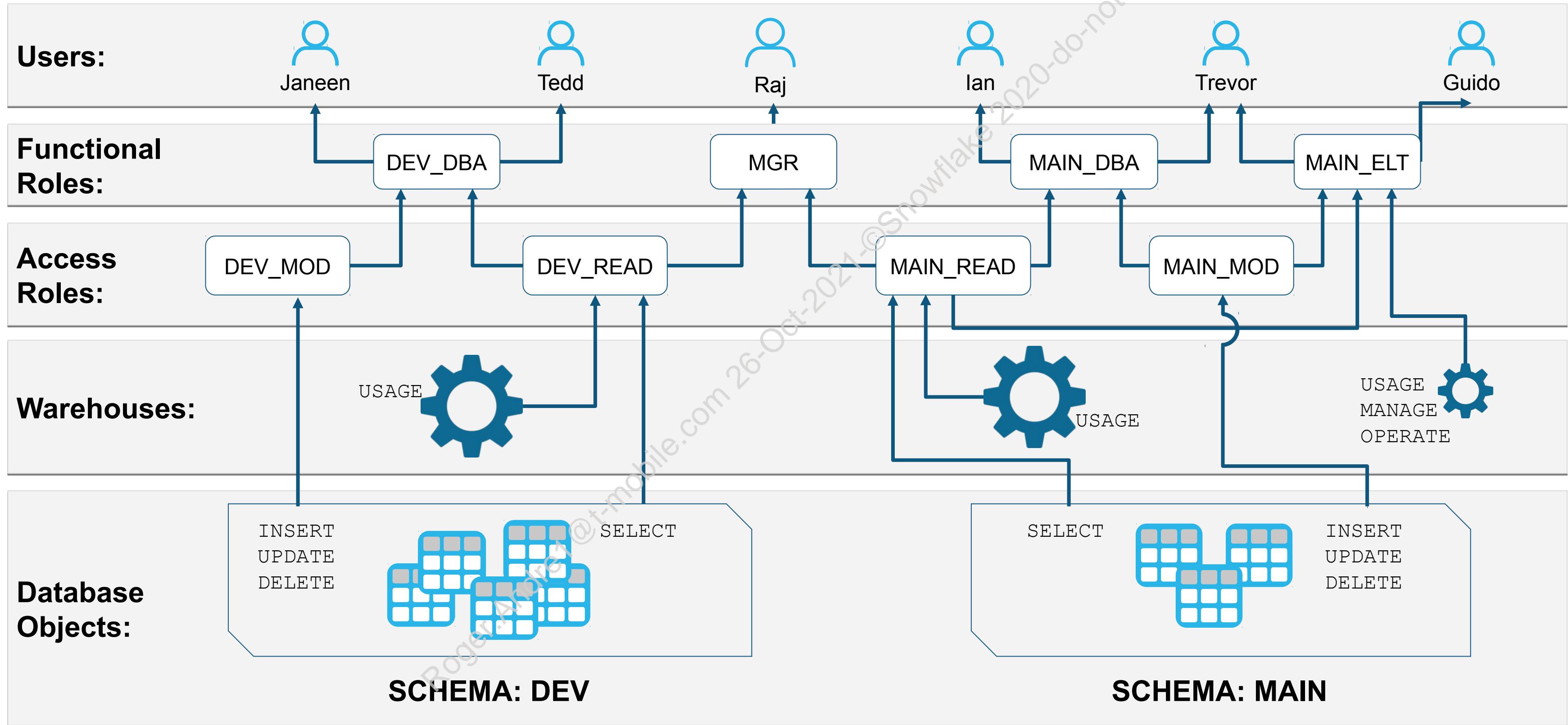
LAYERED ACCESS CONTROL



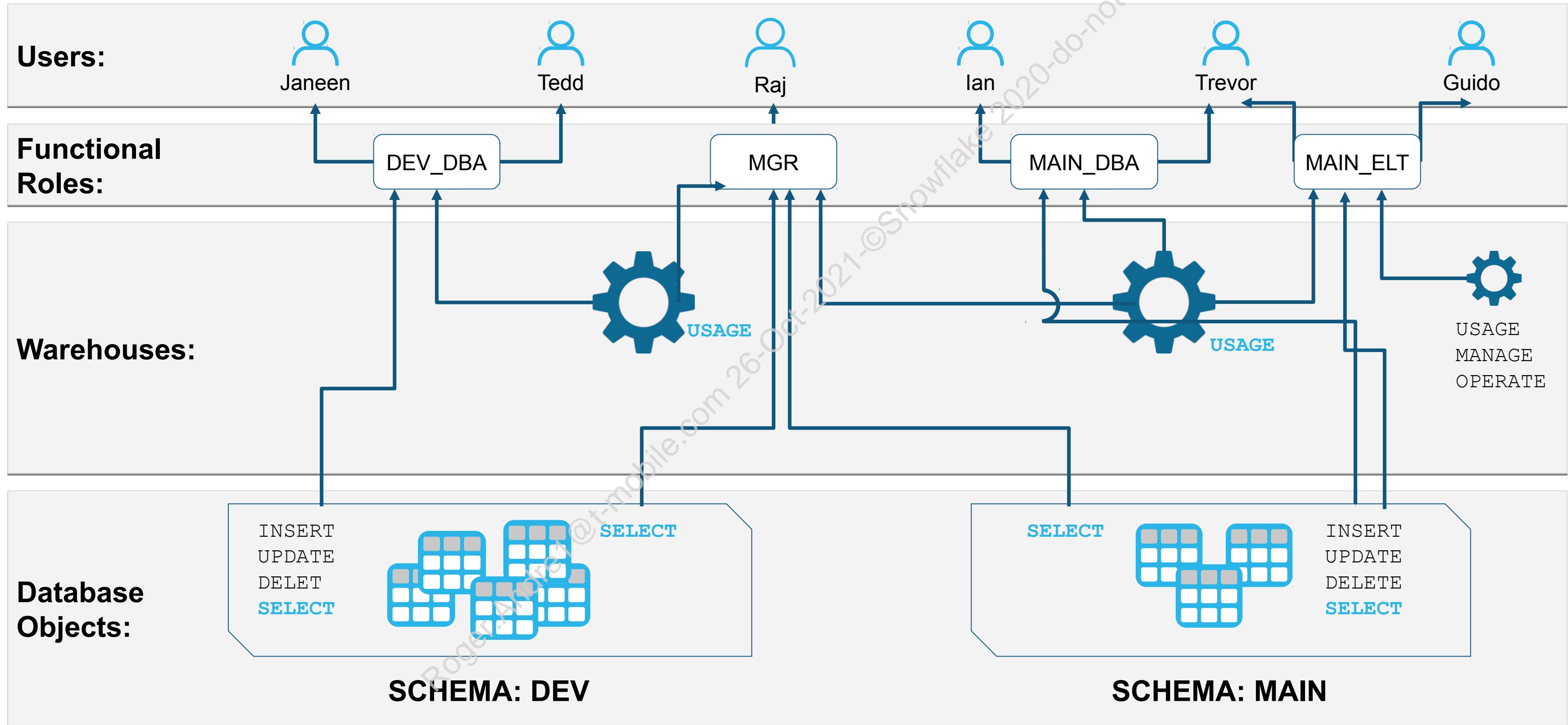
LAYERED ACCESS CONTROL



LAYERED ACCESS CONTROL



LAYERED ACCESS CONTROL



ACCESS MANAGEMENT RECAP

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SCENARIO

FUN WITH ROLES AND PRIVILEGES

- Nancy is new to *CompanyA*, who is using Snowflake
- Their previous Snowflake administrator (Roberto) quit to open a gelato shop in Antarctica, and cannot be reached
- Before leaving, Roberto assigned Nancy to roles ACCOUNTADMIN, SECURITYADMIN, USERADMIN, and SYSADMIN, and left her a coupon for free gelato
- Management has asked Nancy to create a place where new DBA interns can play with their data warehouse without doing any real harm



SCENARIO

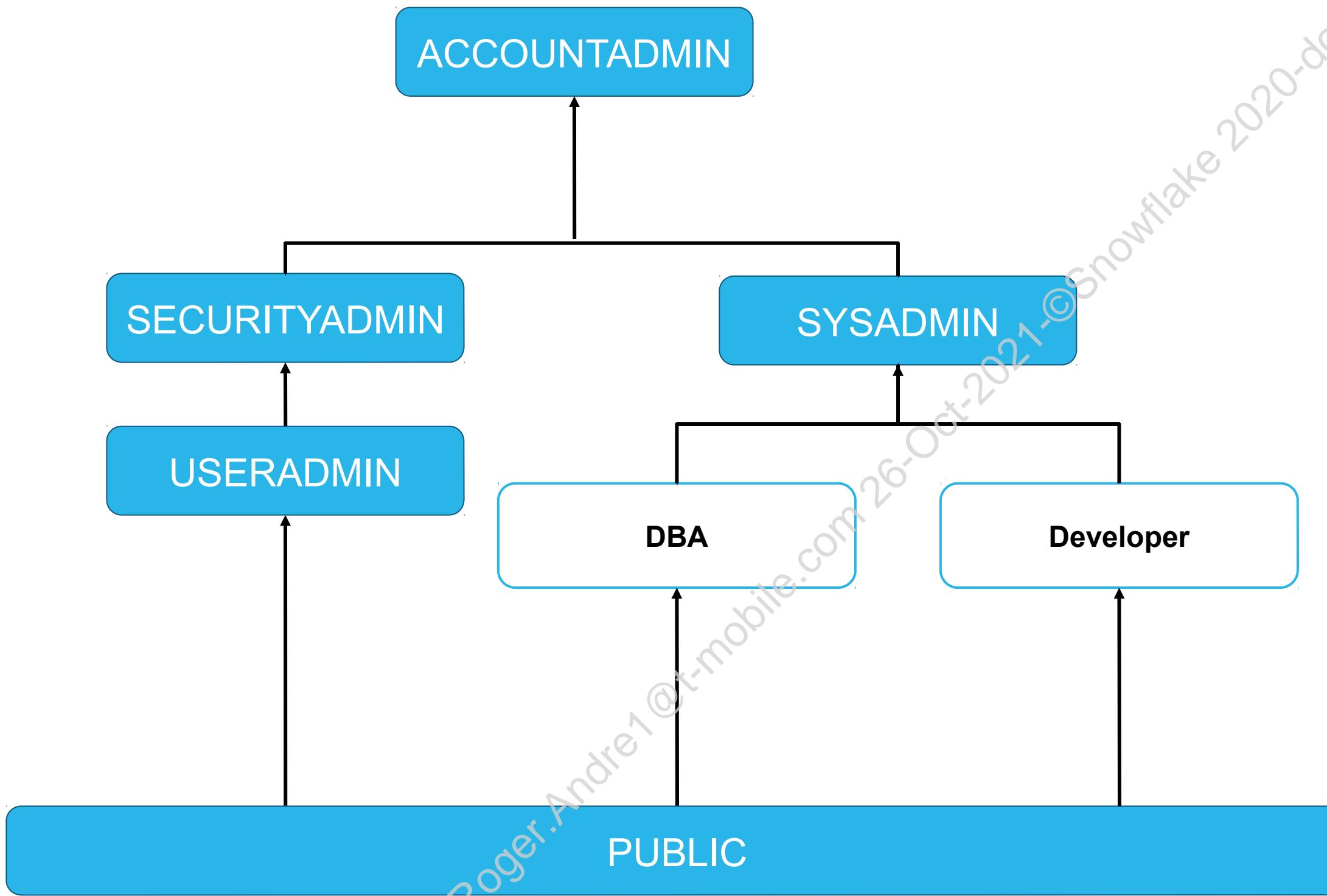
FUN WITH ROLES AND PRIVILEGES

- Create an intern role
- Create an intern_db database
- Give interns the ability to create schemas in the intern_db database

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

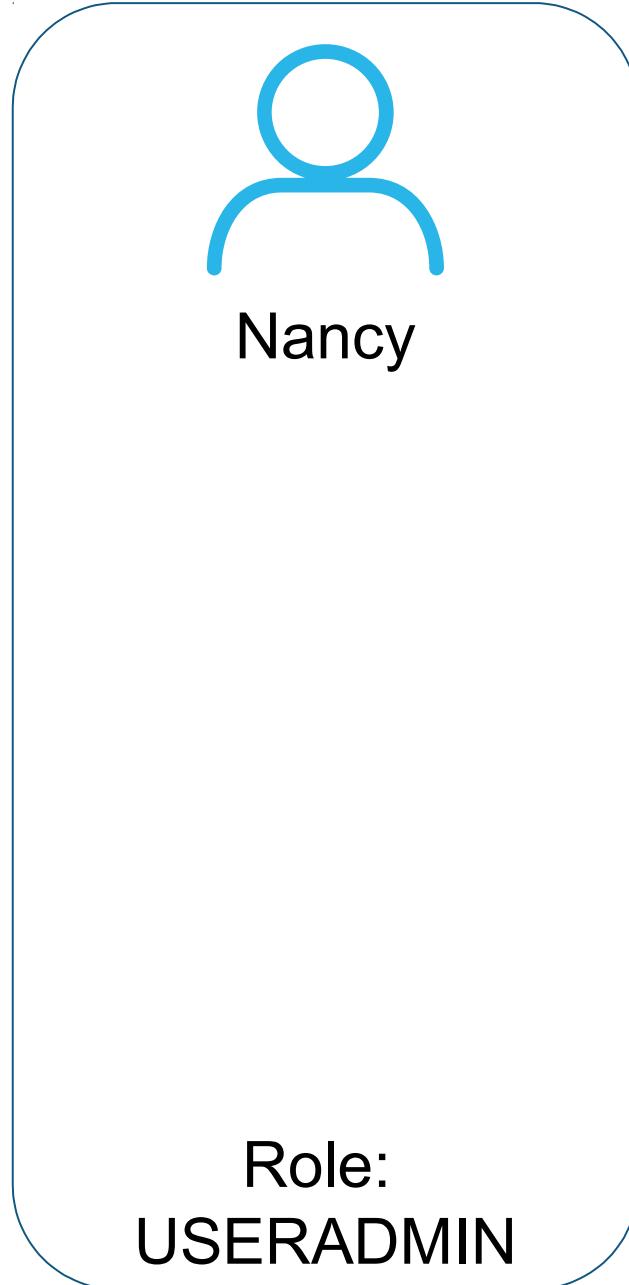


COMPANYA CURRENT CONFIGURATION



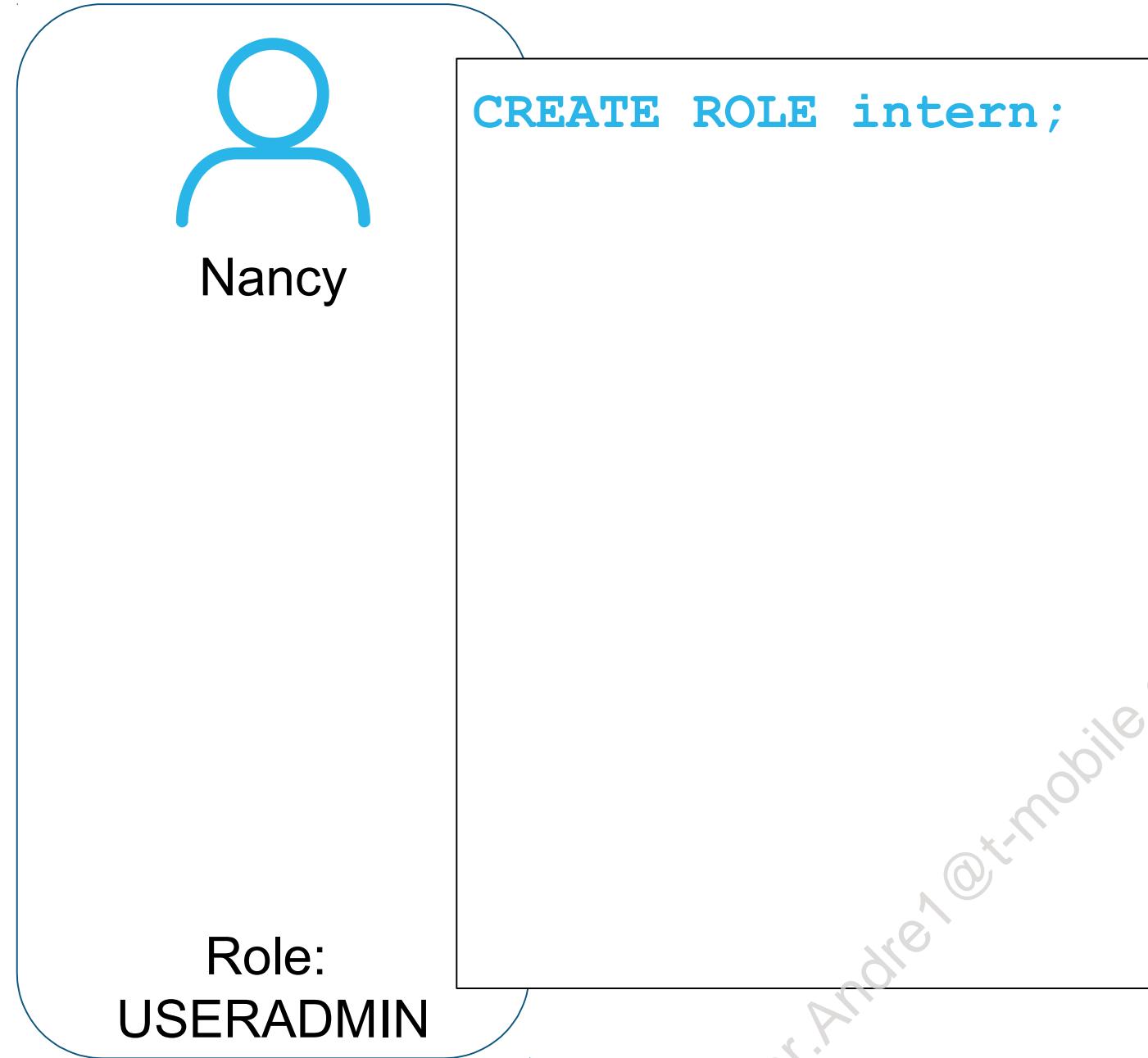
What will Nancy do?

COMPANYA SCENARIO



WE are using the USERADMIN role in this recap. The SECURITYADMIN role would also work.

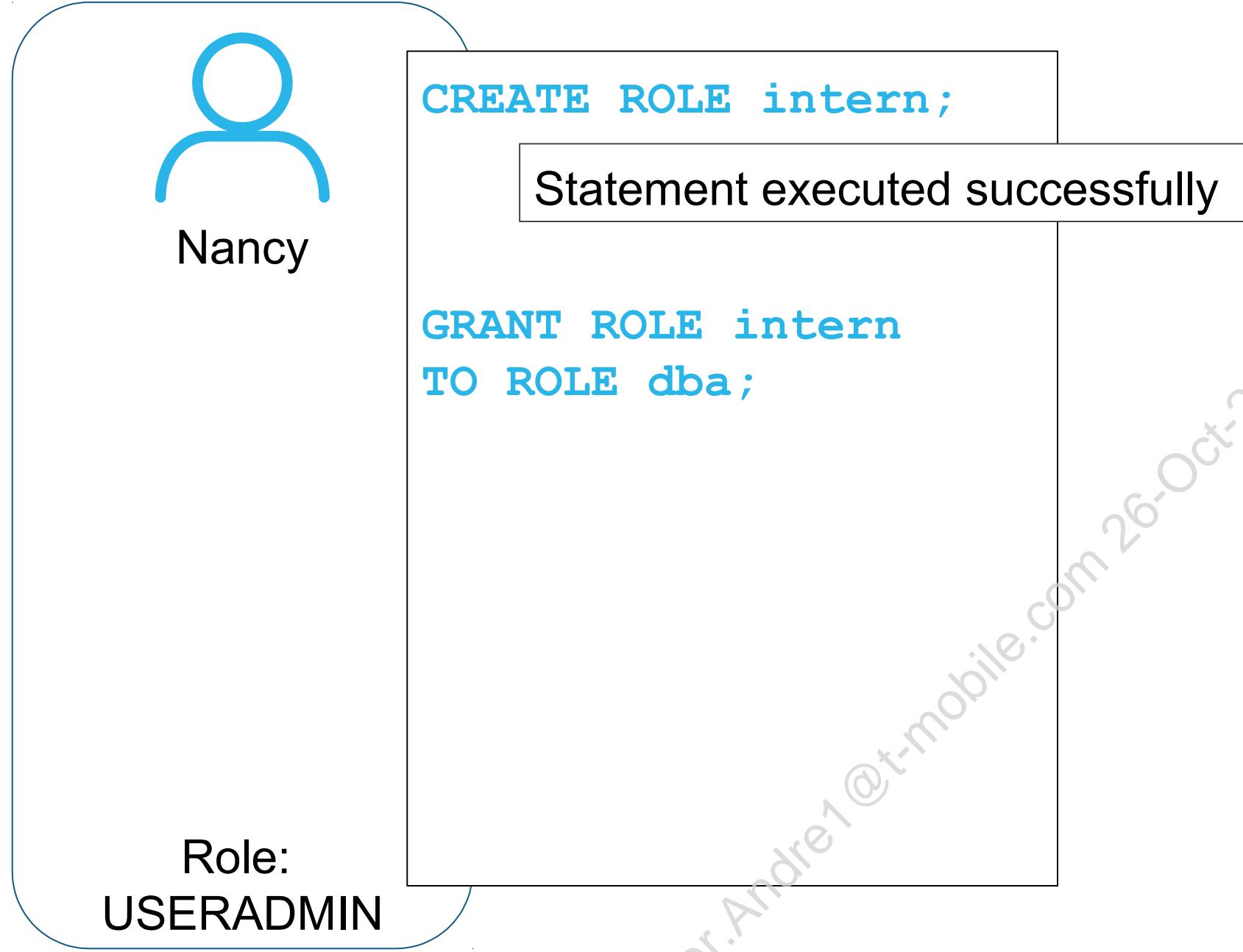
COMPANYA SCENARIO



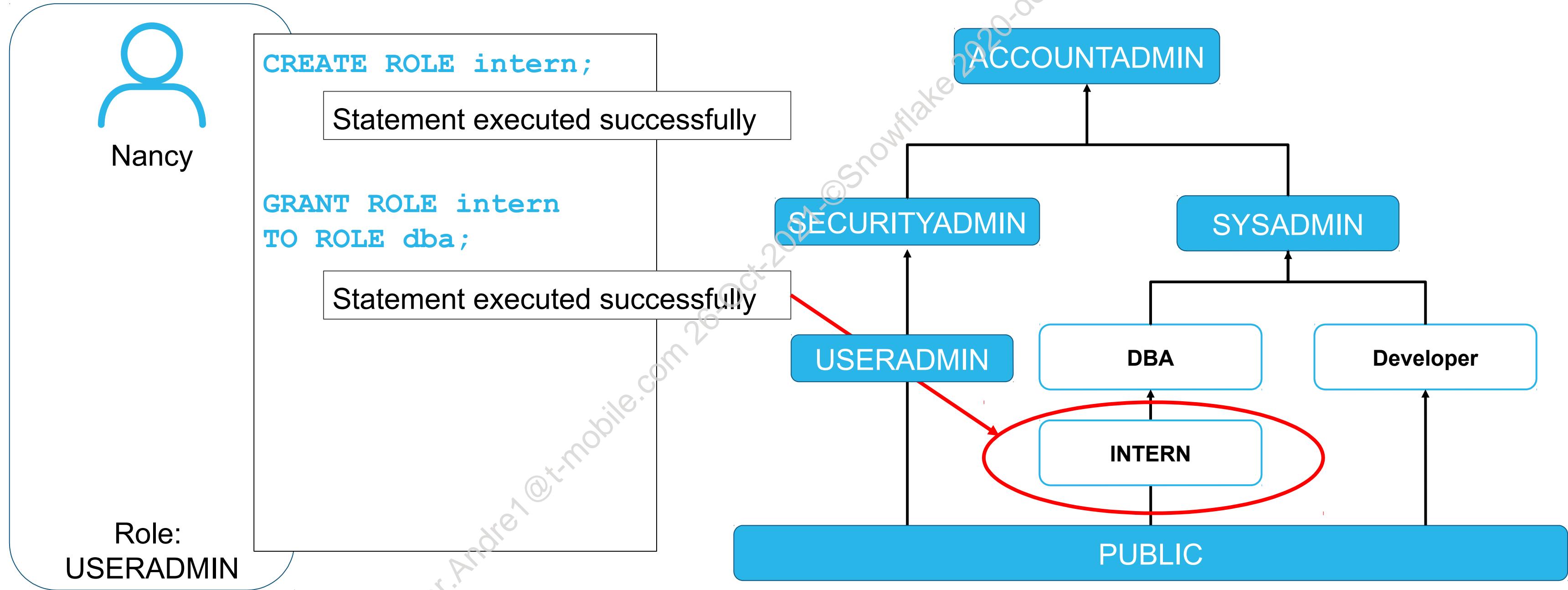
Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



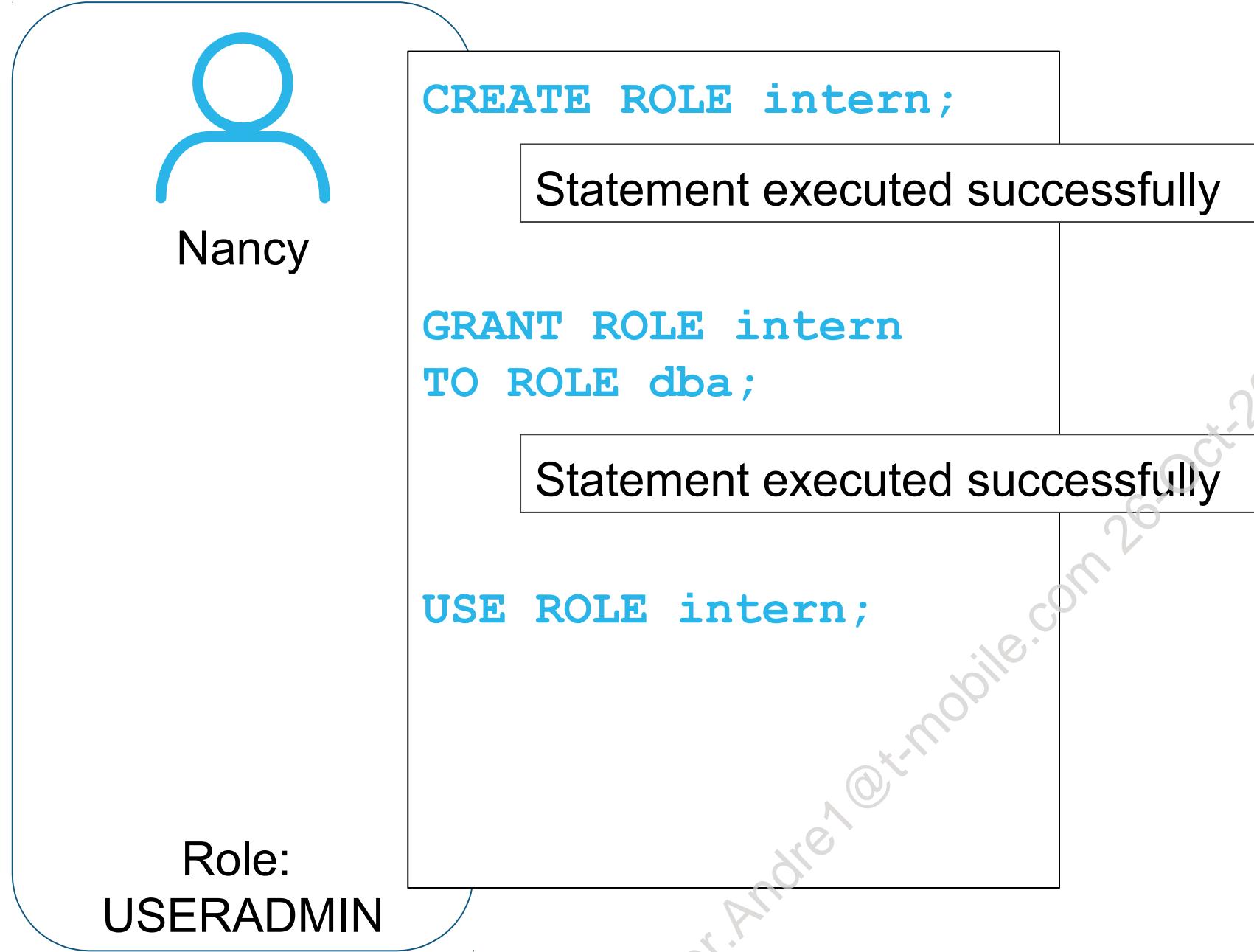
COMPANYA SCENARIO



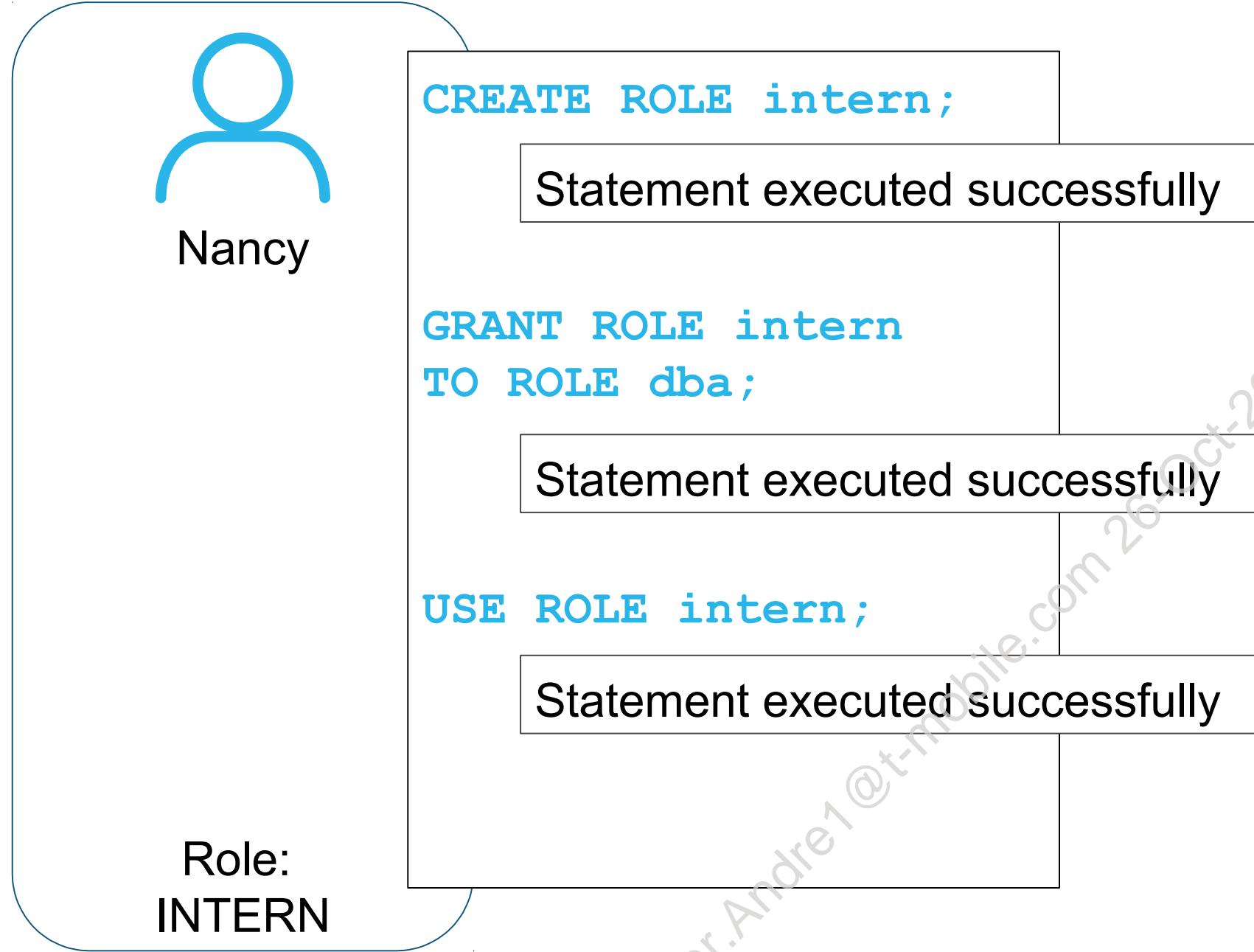
COMPANYA SCENARIO



COMPANYA SCENARIO

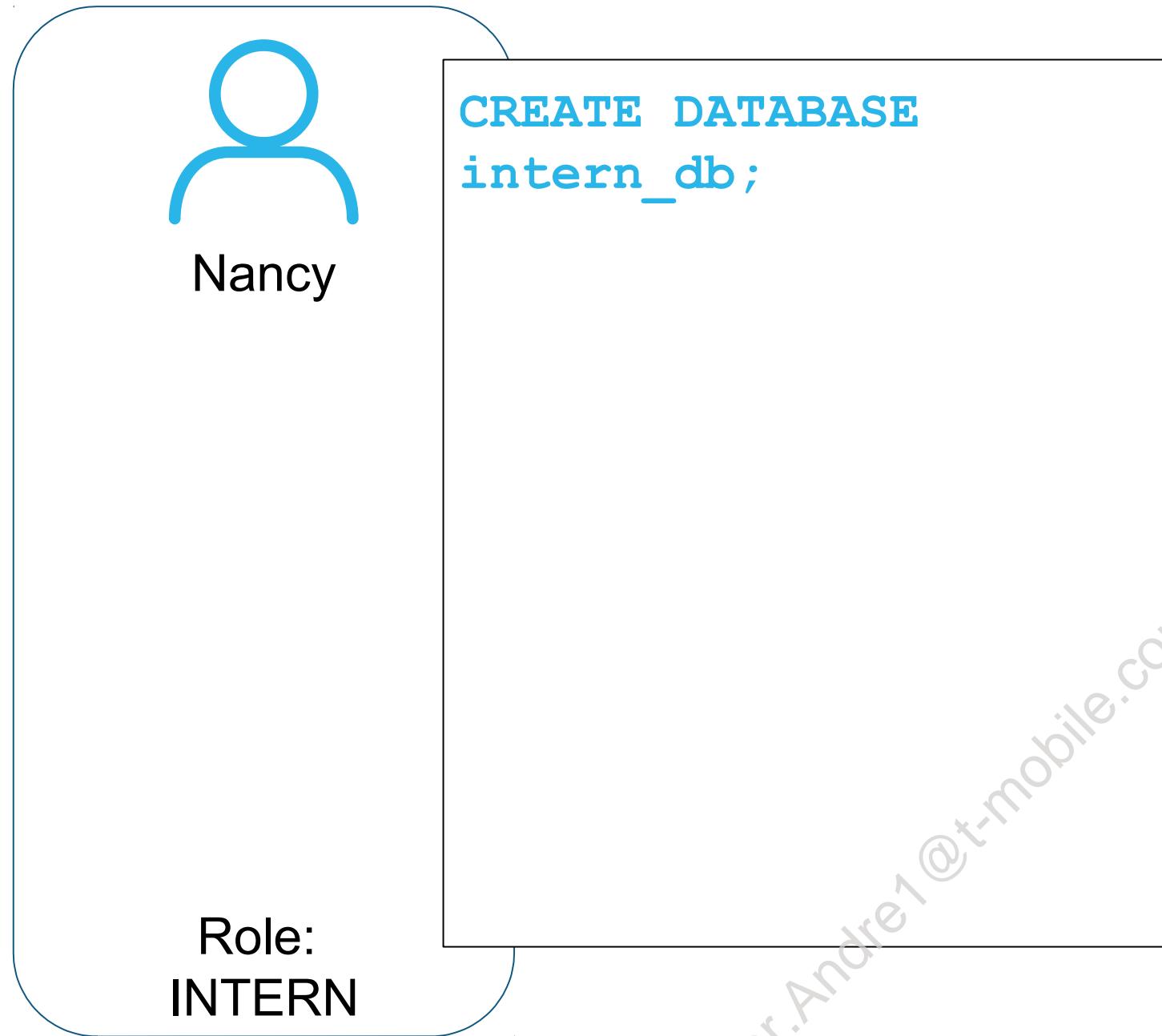


COMPANYA SCENARIO

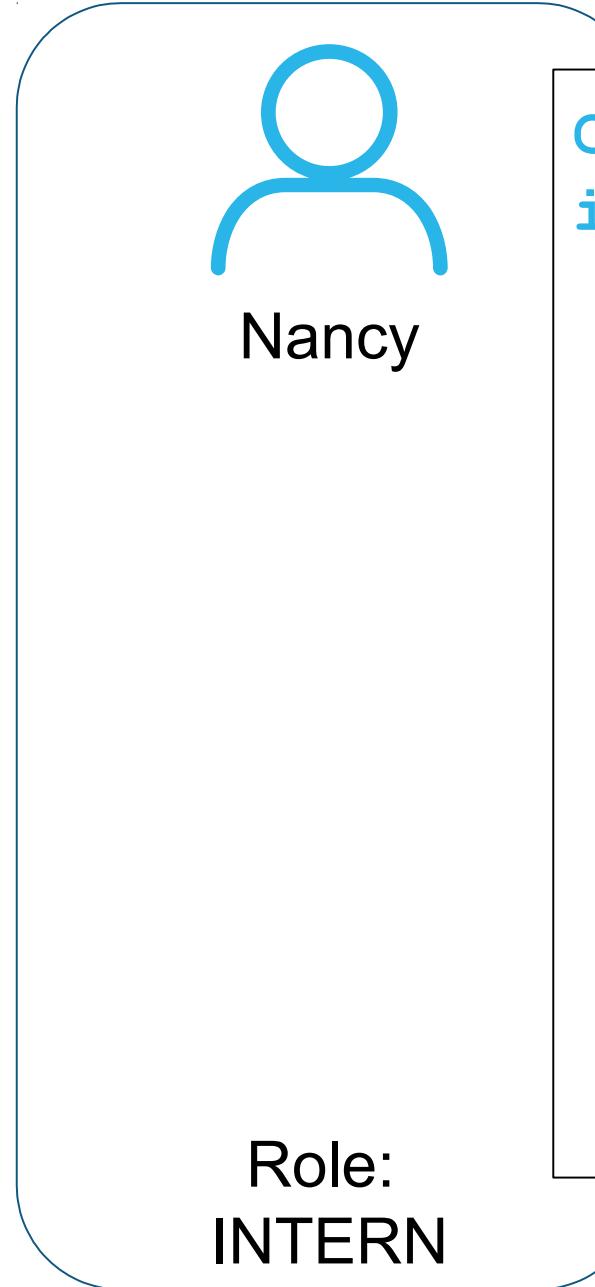


This will work since the SYSADMIN role has been granted to Nancy and the intern role is inline with the SYSADMIN role.

COMPANYA SCENARIO

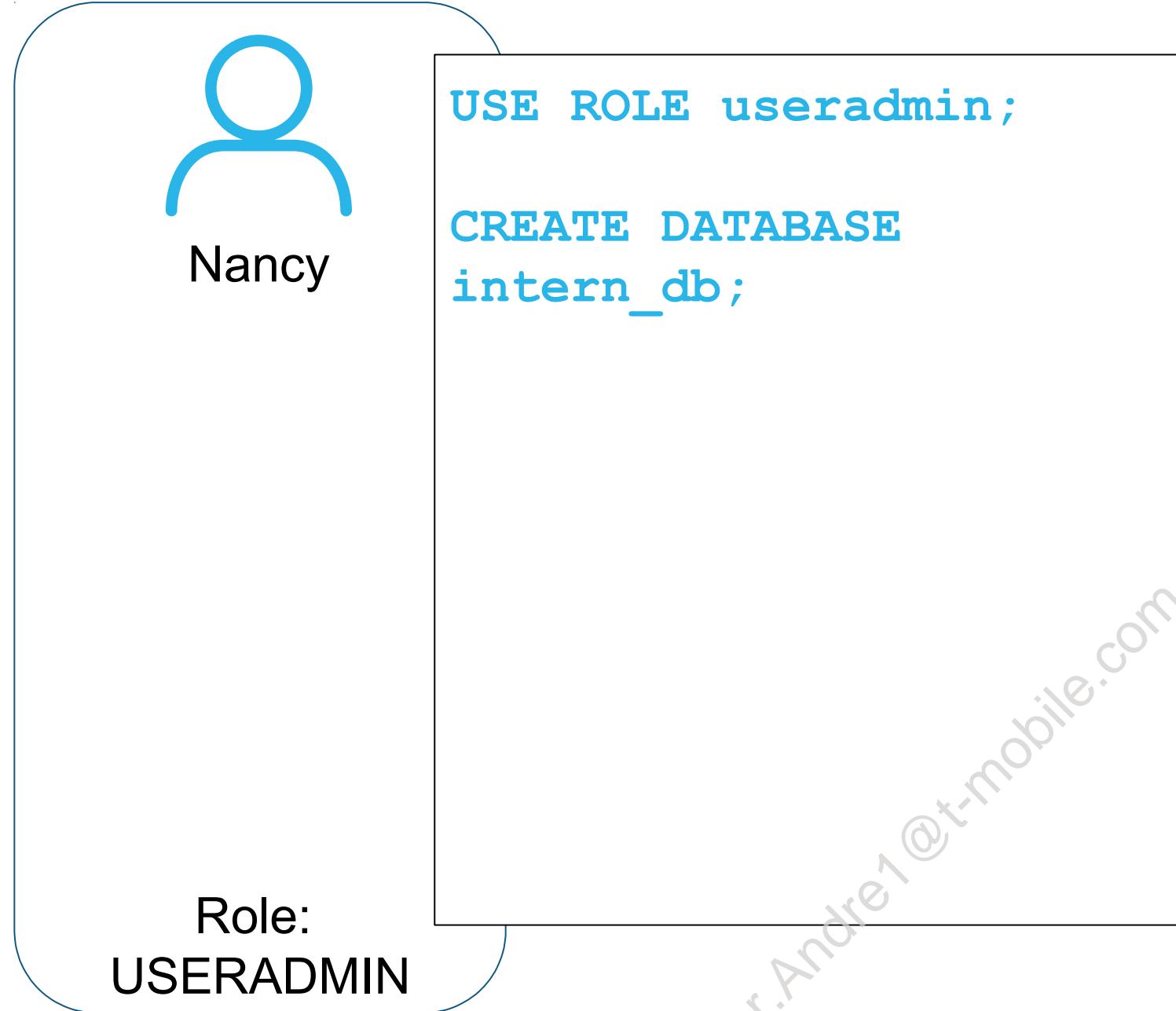


COMPANYA SCENARIO

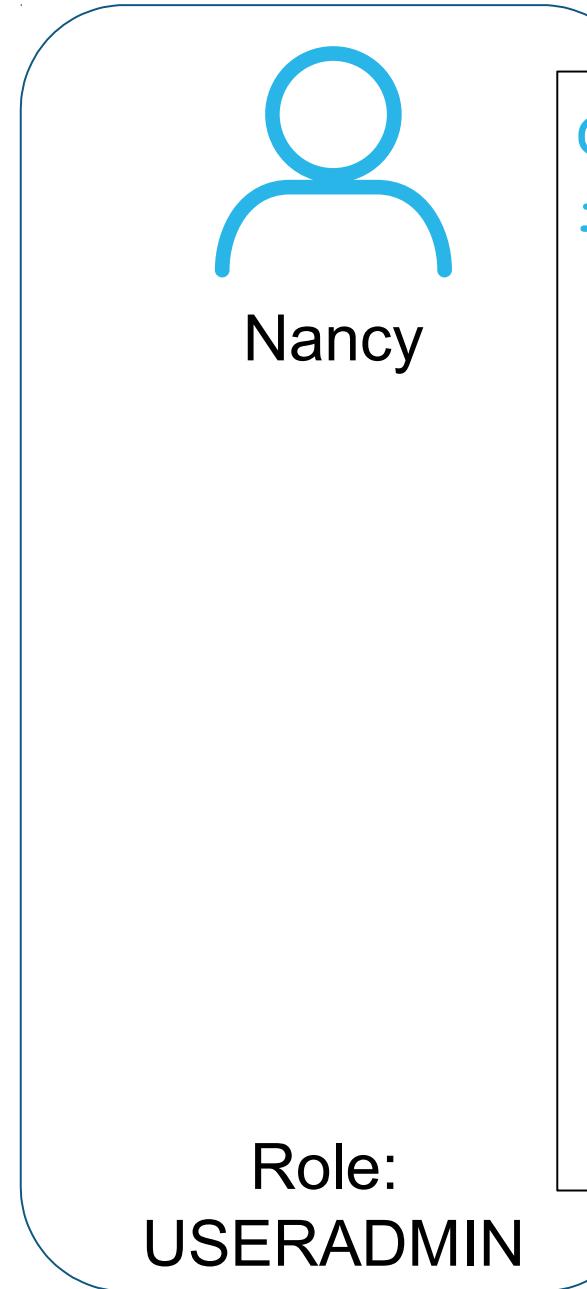


The role INTERN has not been granted the privilege to create a database. Even though it's in the same hierarchy as the SYSADMIN role, the permissions doesn't transfer down.

COMPANYA SCENARIO

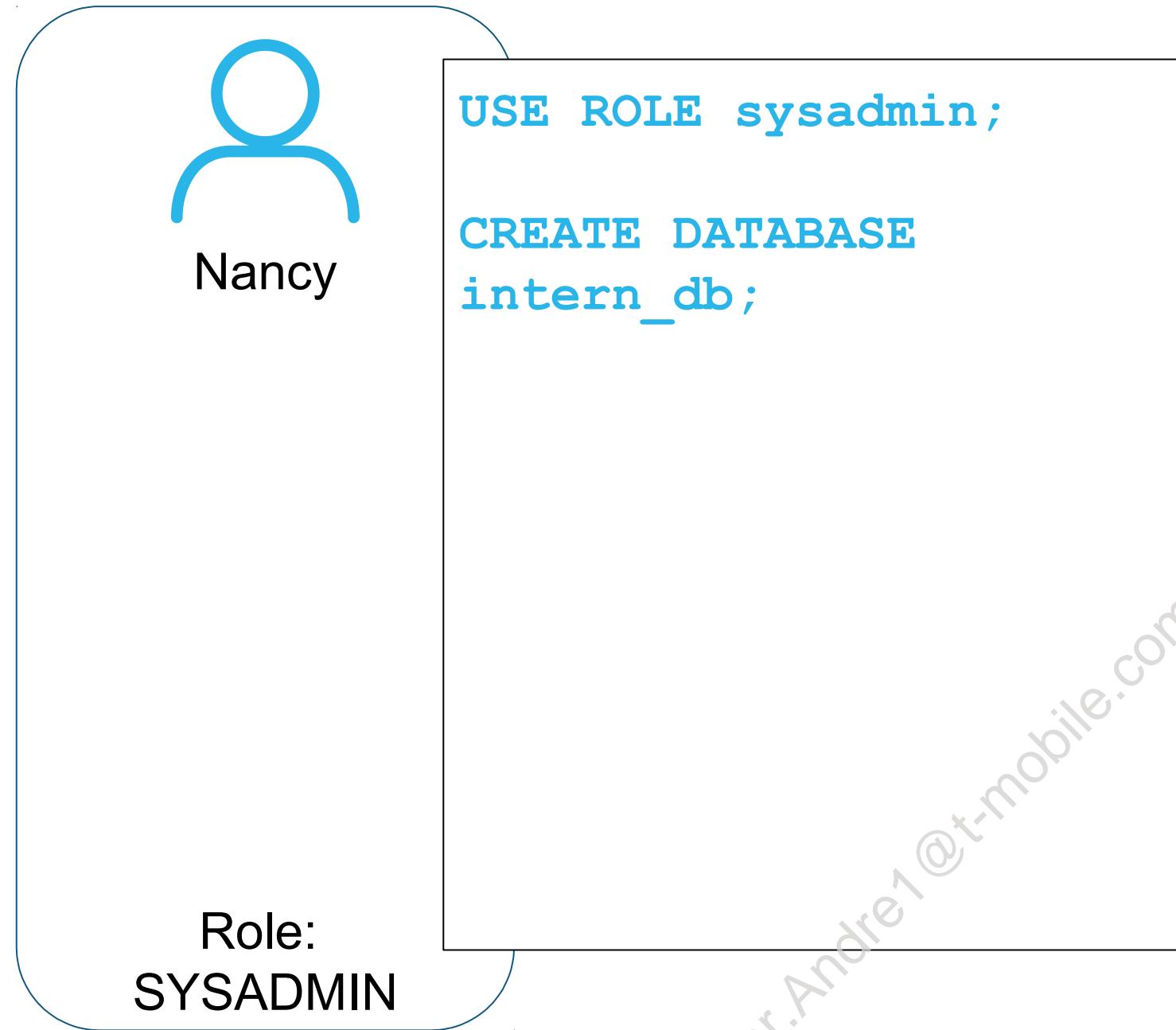


COMPANYA SCENARIO

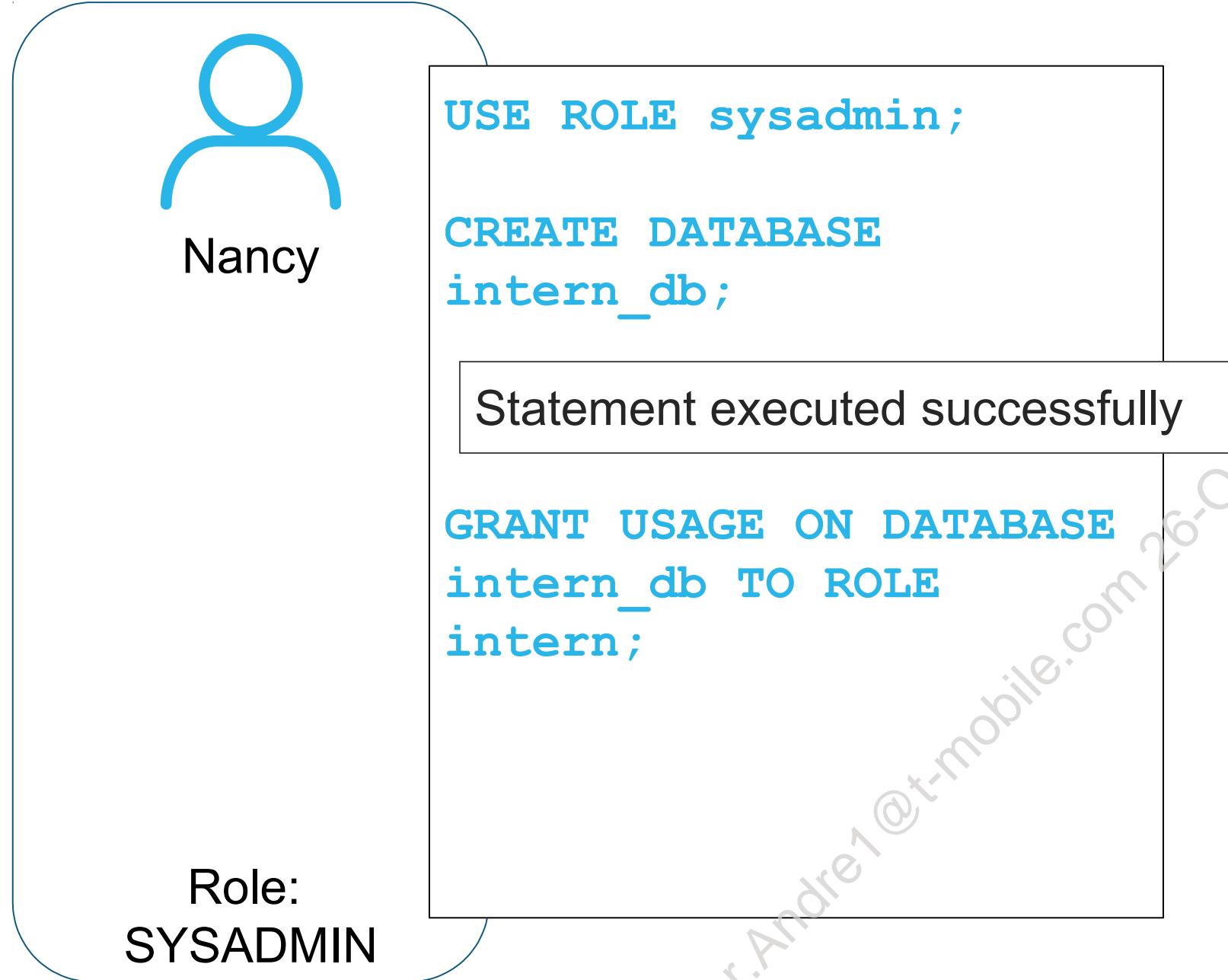


The role USERADMIN does not have the ability to create objects – only grant privileges

COMPANYA SCENARIO

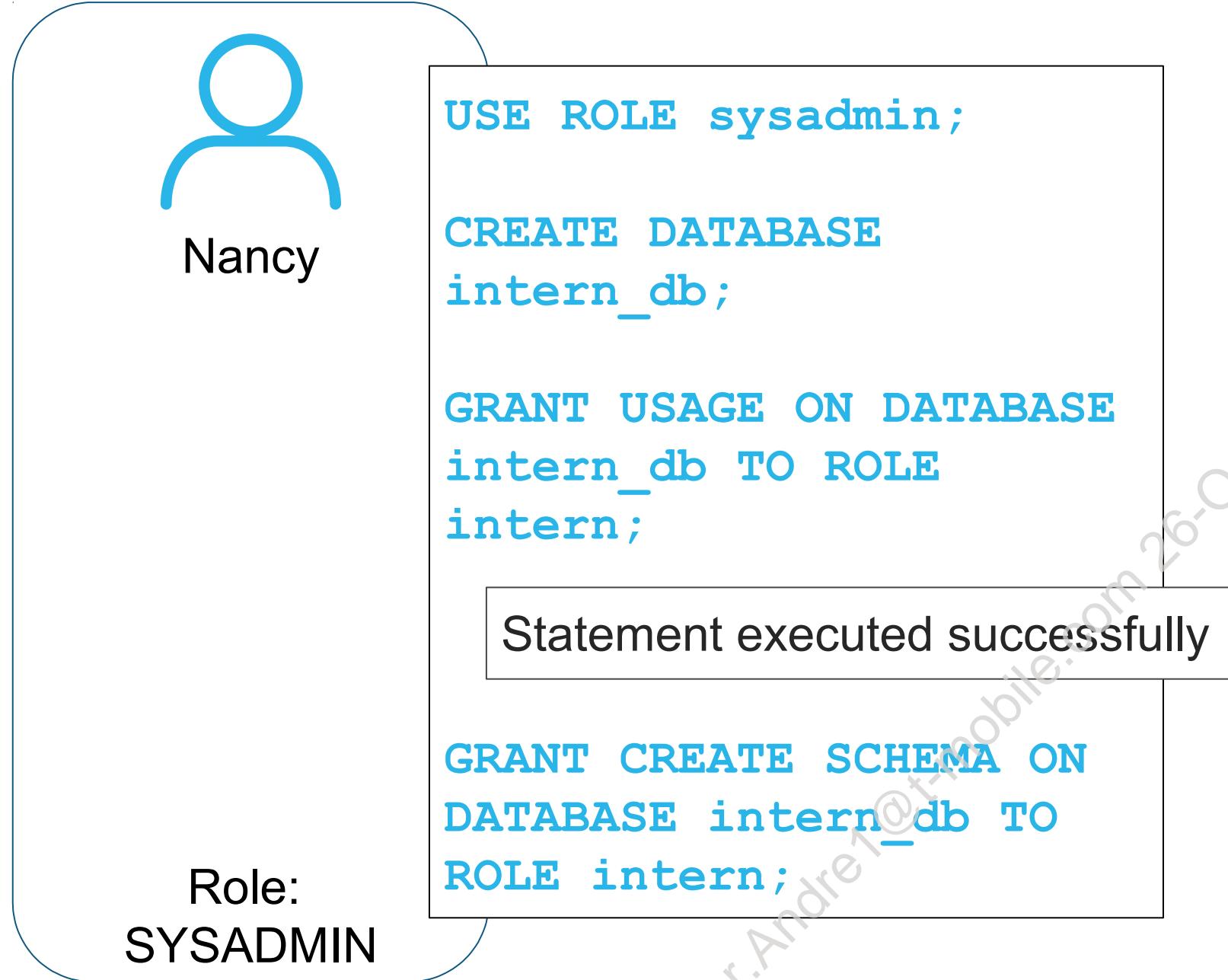


COMPANYA SCENARIO



The role SYSADMIN has the ability to create objects

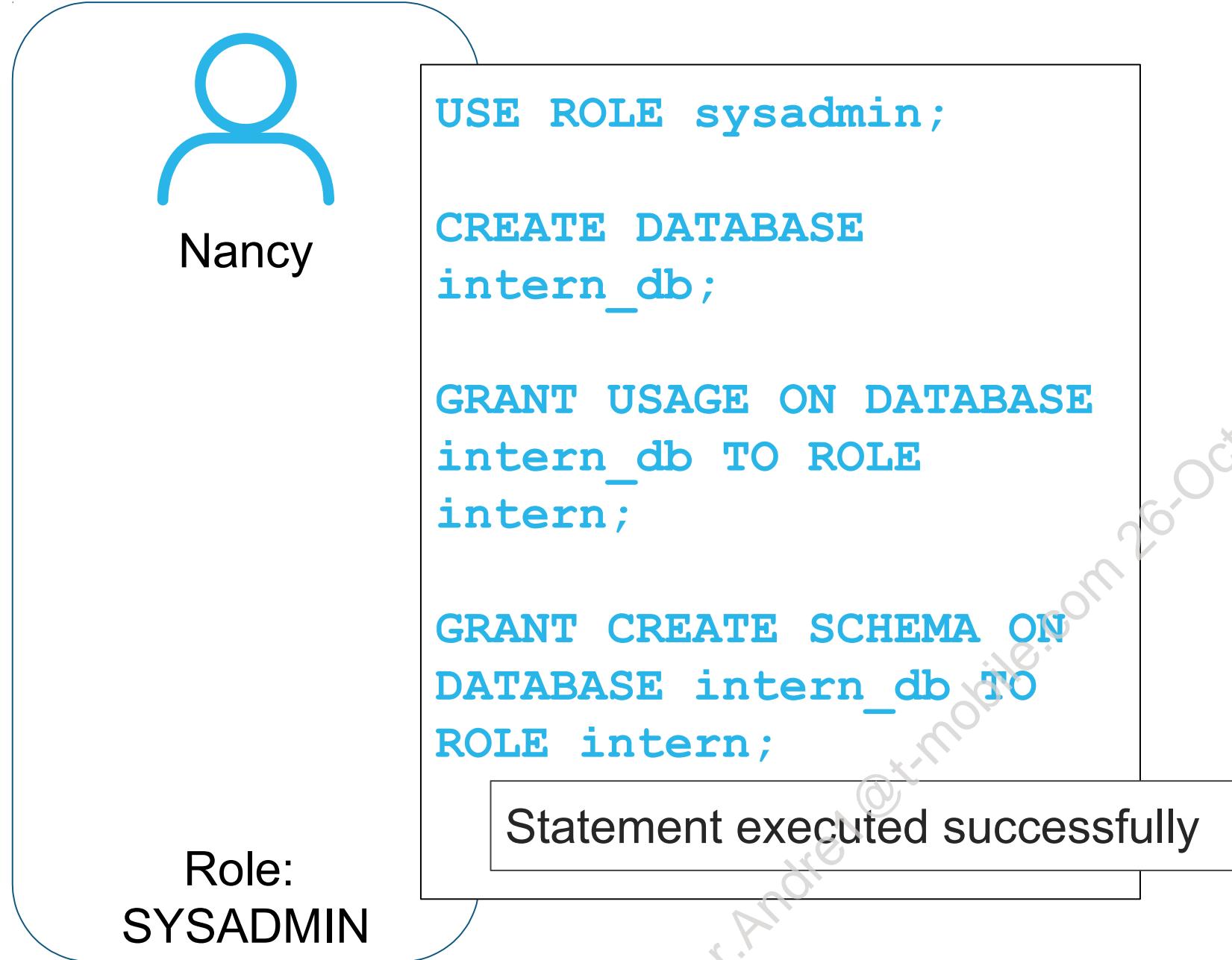
COMPANYA SCENARIO



The role **SYSADMIN** has the ability to create objects

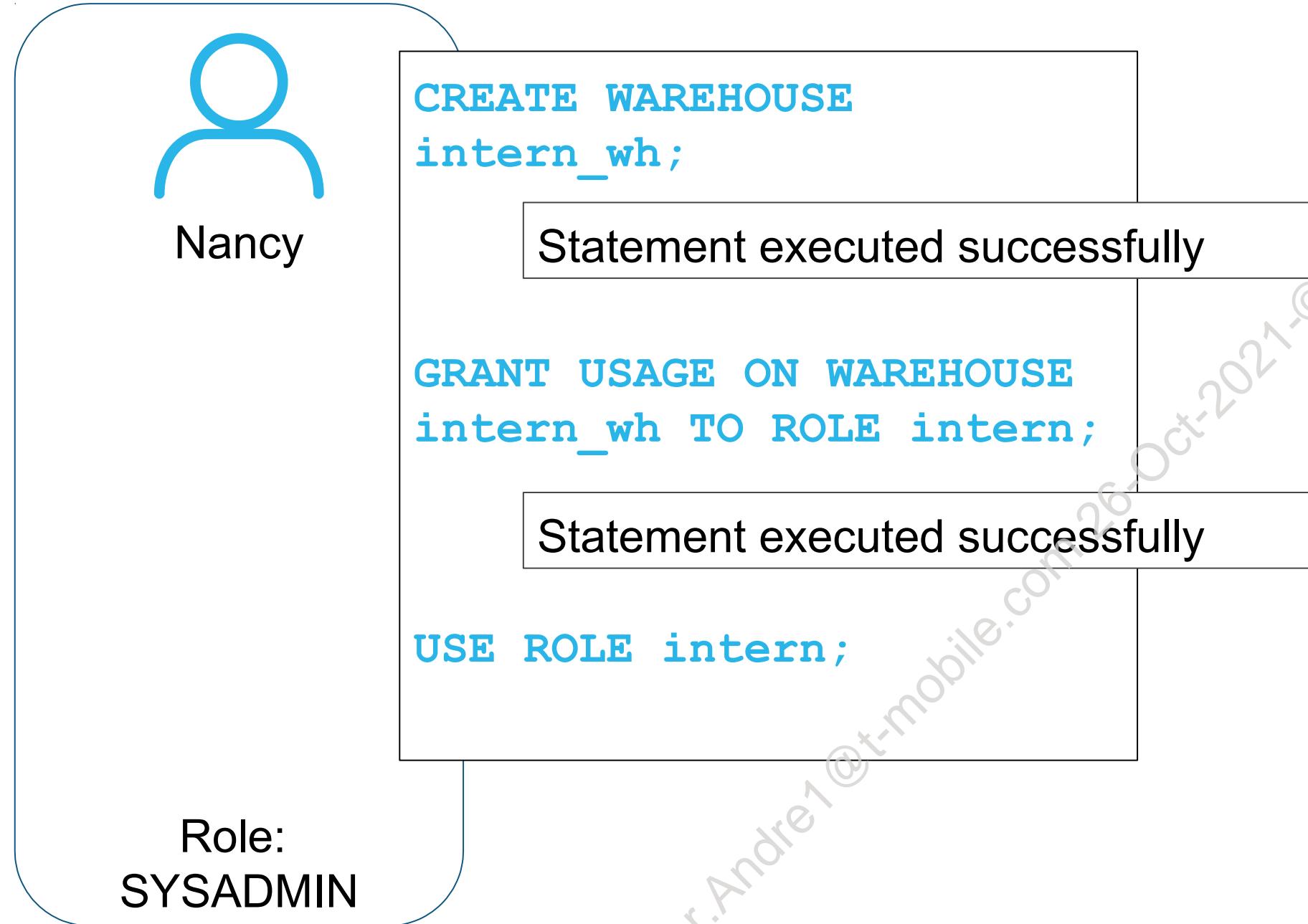


COMPANYA SCENARIO



The role SYSADMIN has the ability to create objects

COMPANYA SCENARIO



While she's at it, Nancy creates a warehouse for the interns, and grants them USAGE on it.

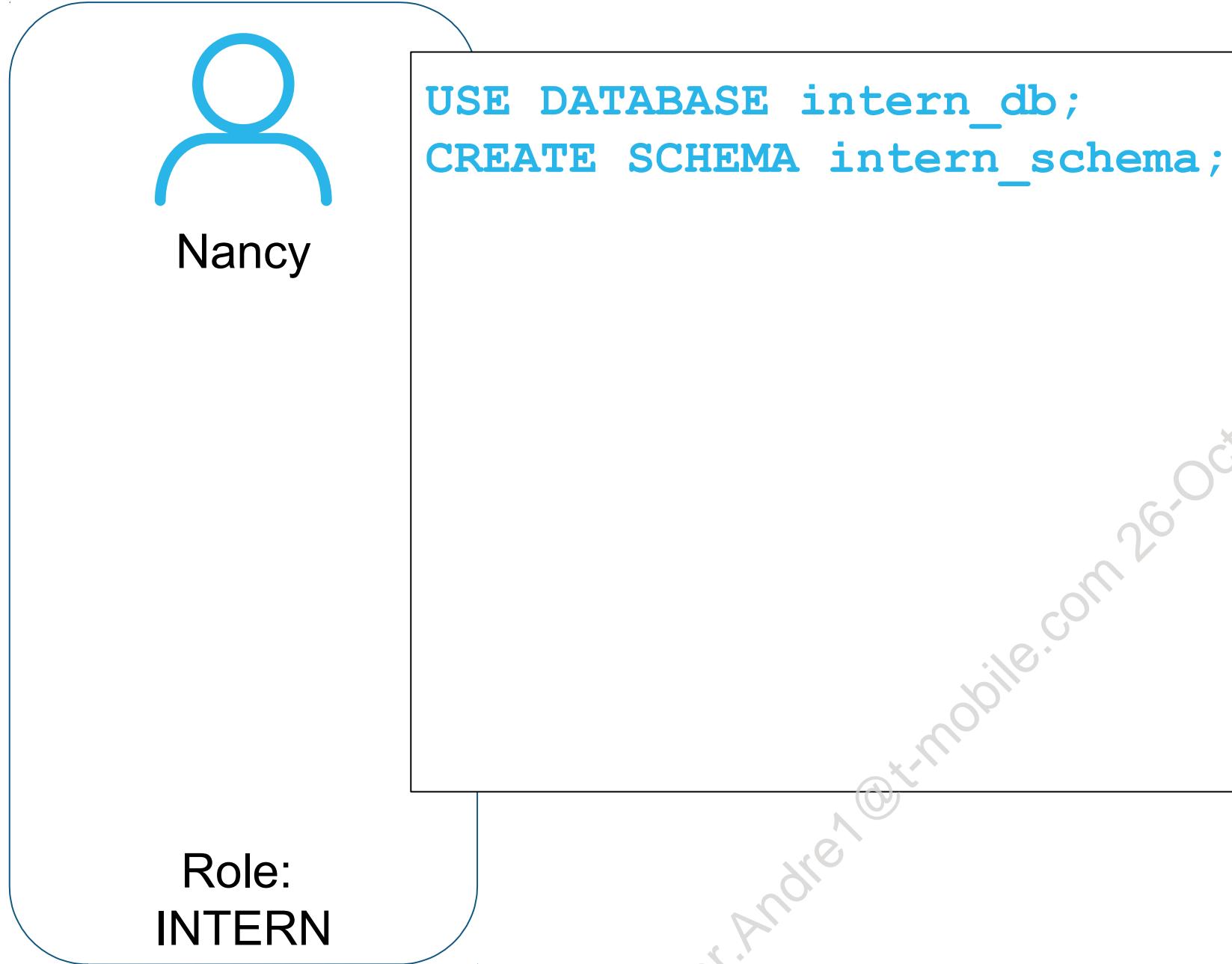
INTERN

Permissions:

- USAGE on DATABASE intern
- CREATE SCHEMA on DATABASE intern
- USAGE ON intern_wh



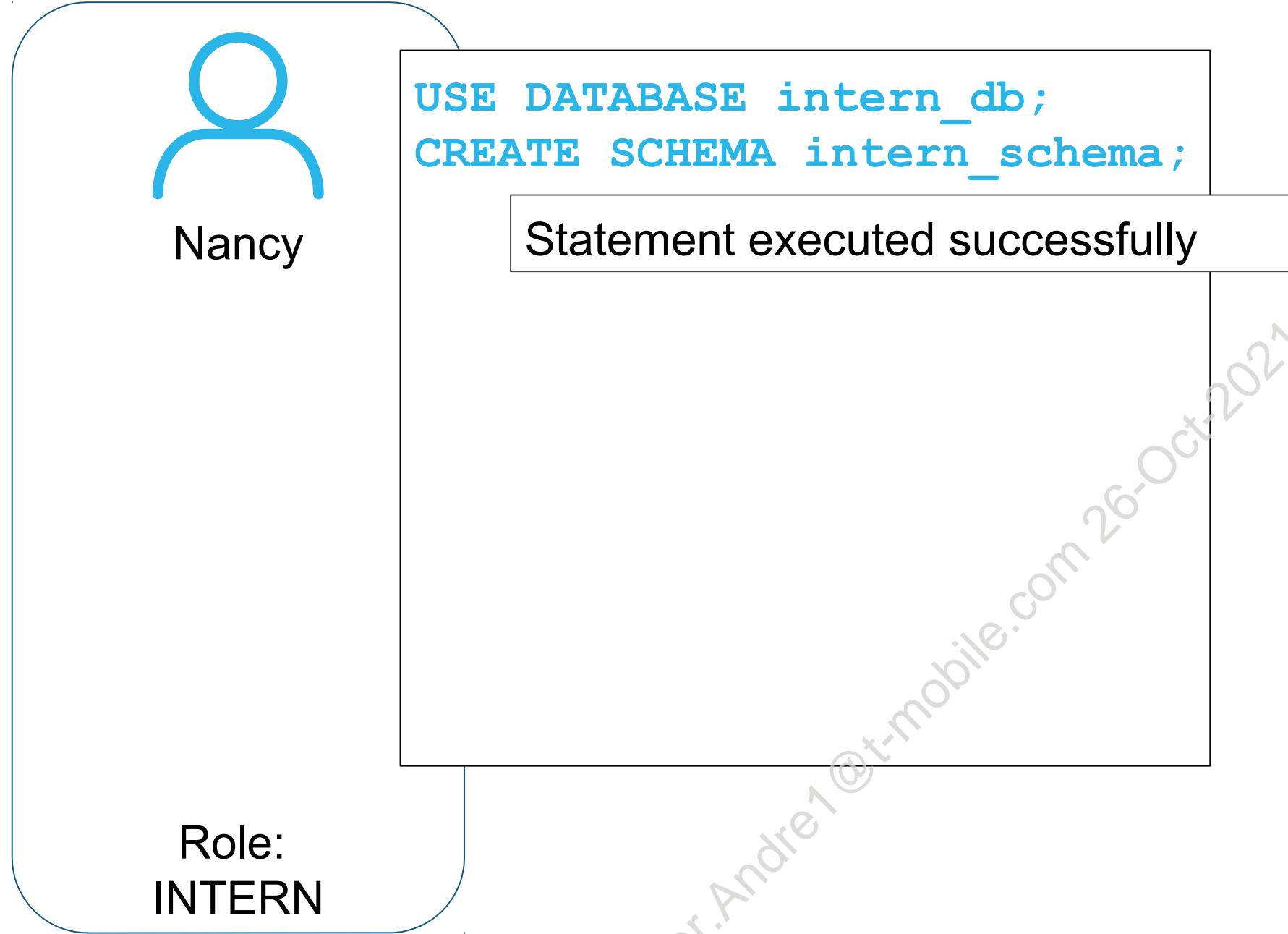
COMPANYA SCENARIO



Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



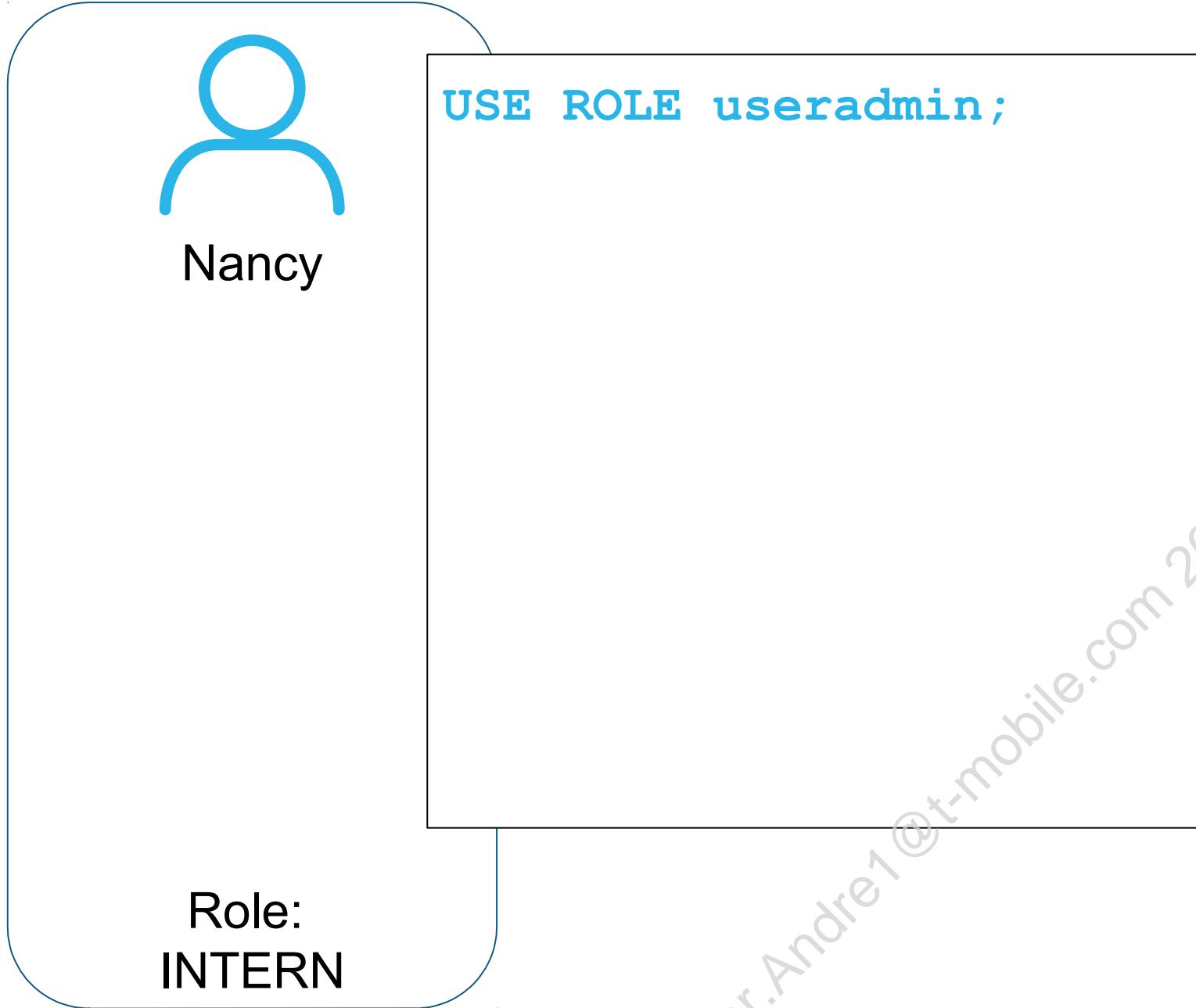
COMPANYA SCENARIO



Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



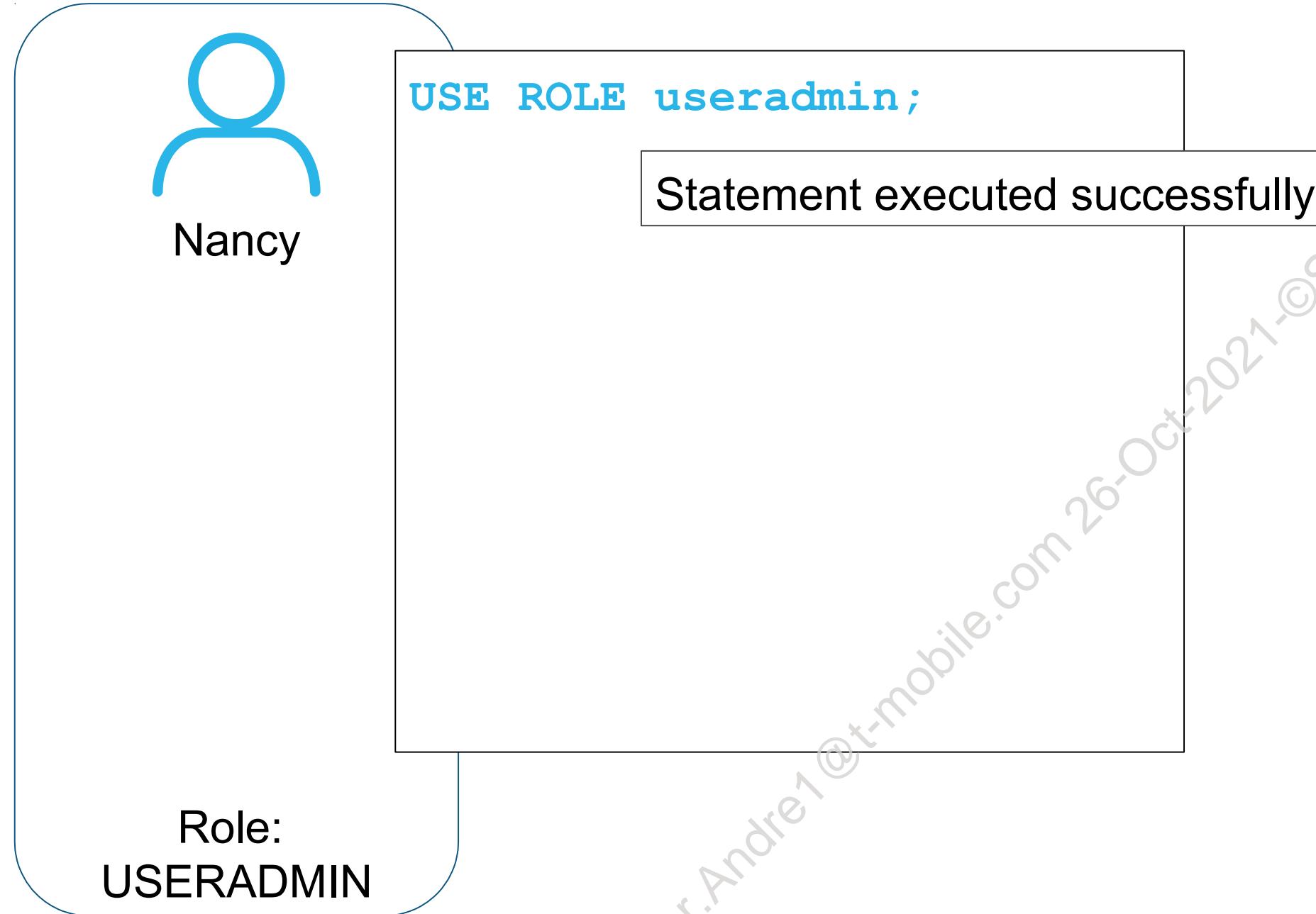
COMPANYA SCENARIO



Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



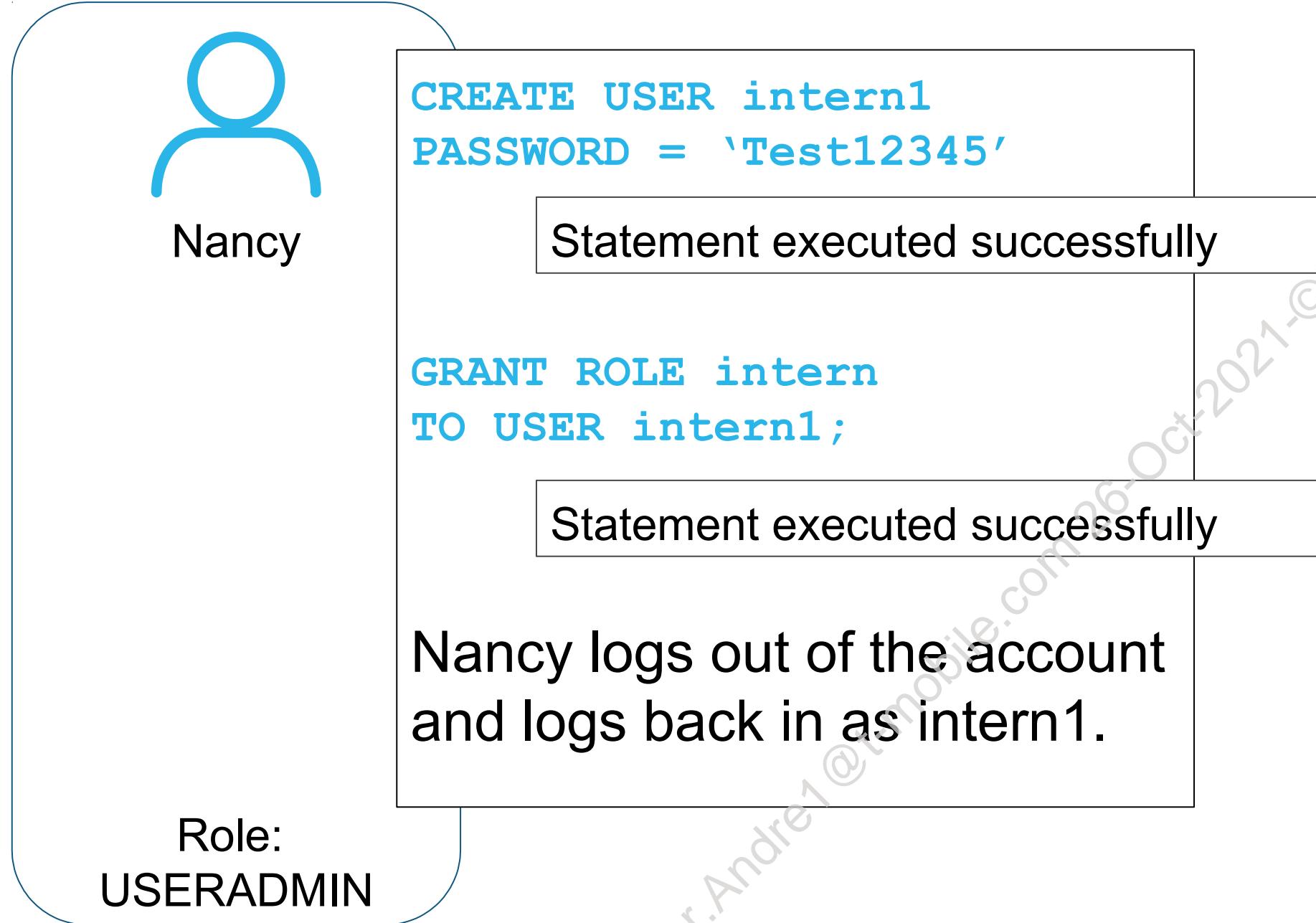
COMPANYA SCENARIO



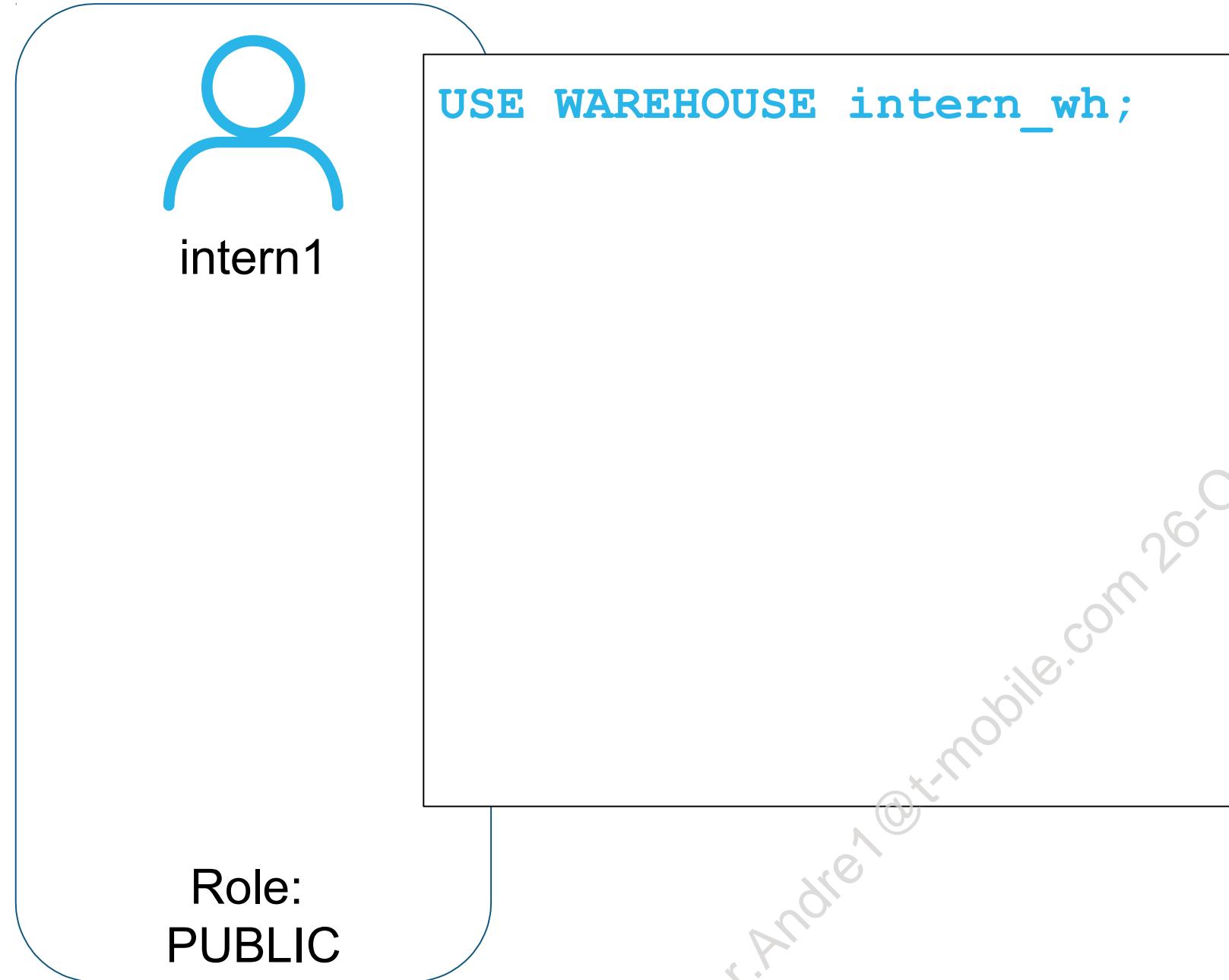
This is not an intern becoming a USERADMIN – this is Nancy using a role she has been granted.

Other users in the INTERN role would not be able to do this.

COMPANYA SCENARIO

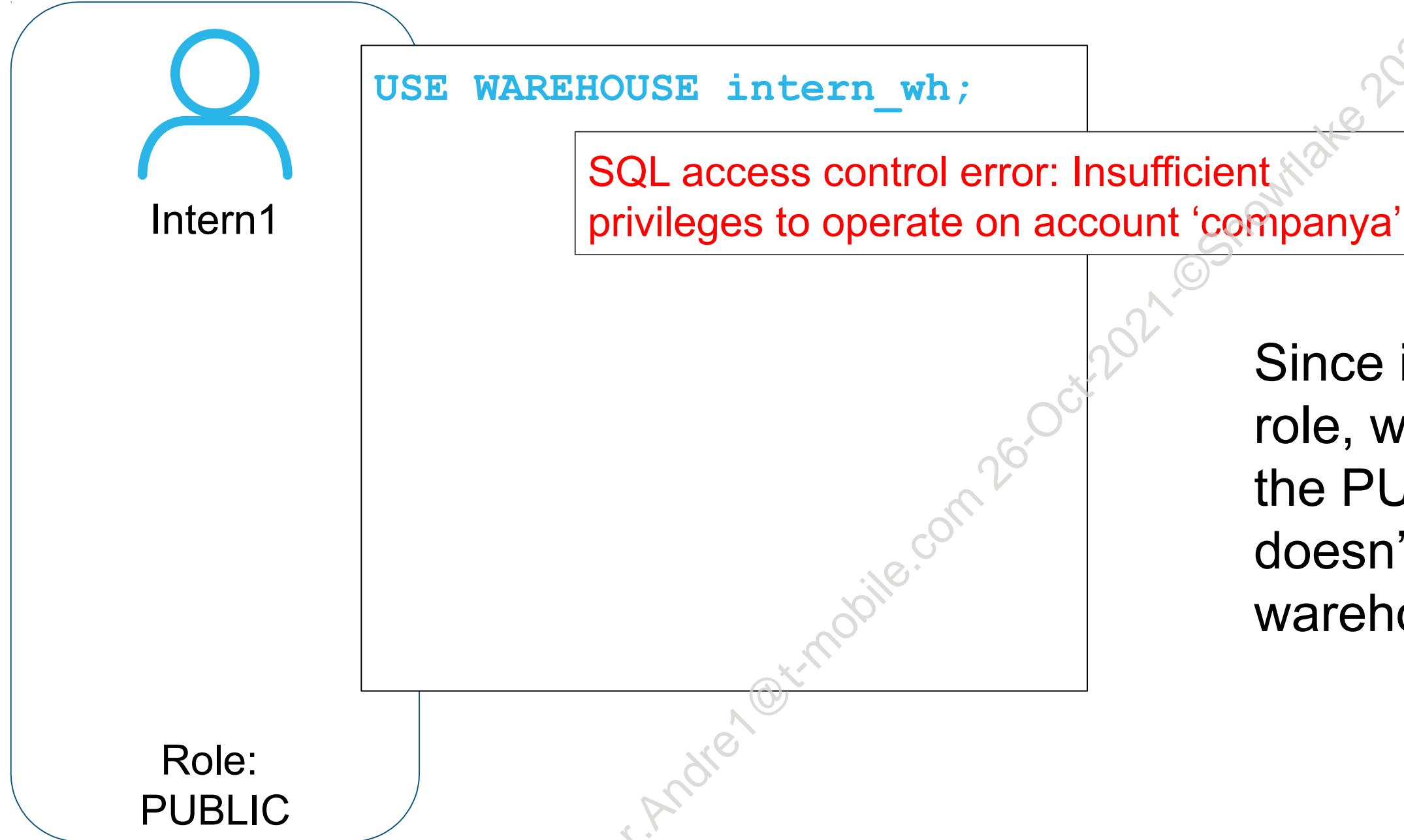


COMPANYA SCENARIO



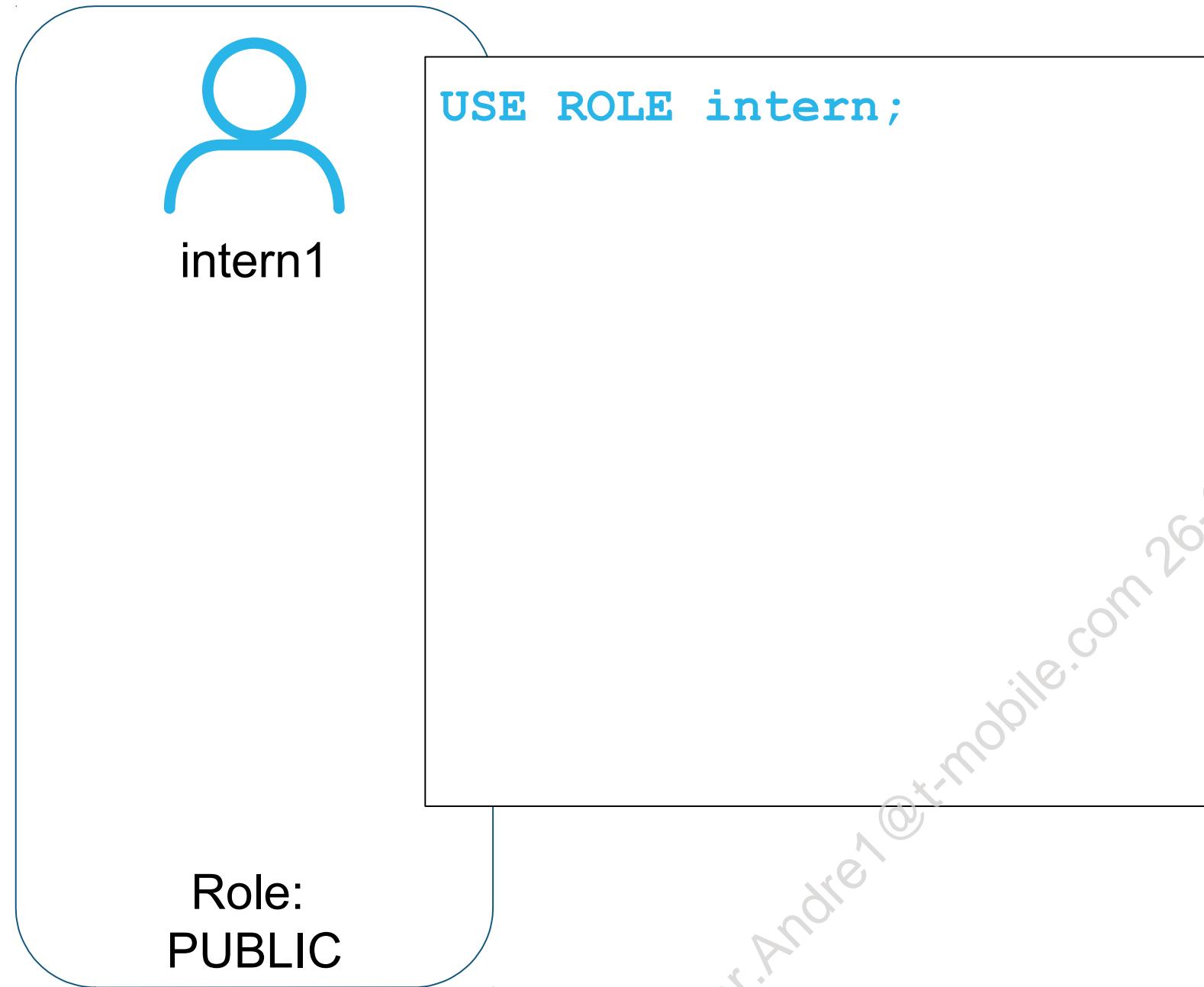
Since **intern1** is not assigned a default role, it will be assigned to the public role by default.

COMPANYA SCENARIO



Since intern1 does not have a default role, when they log in, they will be in the PUBLIC role. The PUBLIC role doesn't have access to the `intern_wh` warehouse.

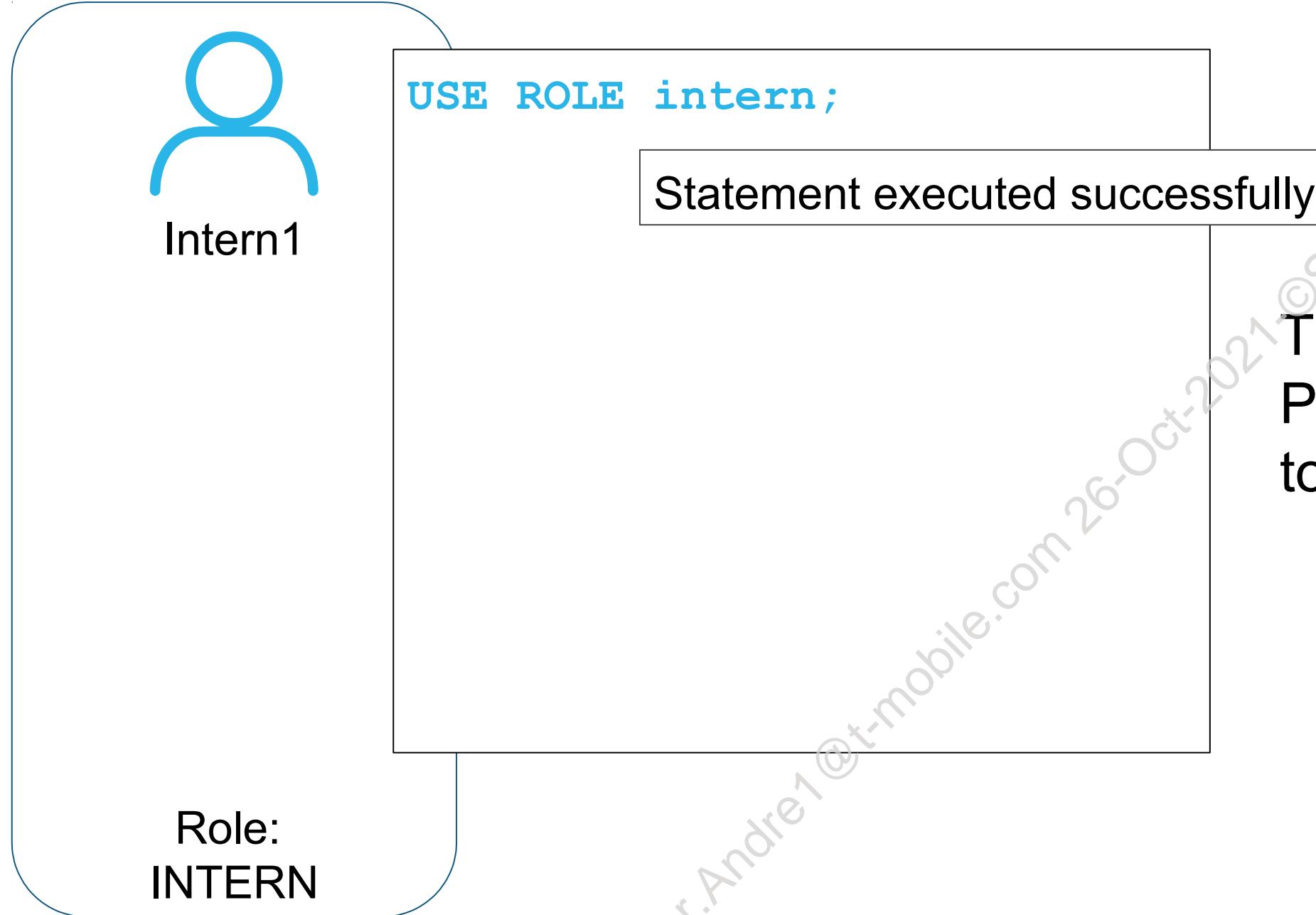
COMPANYA SCENARIO



Roger.Andrei1@t-mobile.com 26-Oct-2021-©Snowflake 2020-do-not-copy

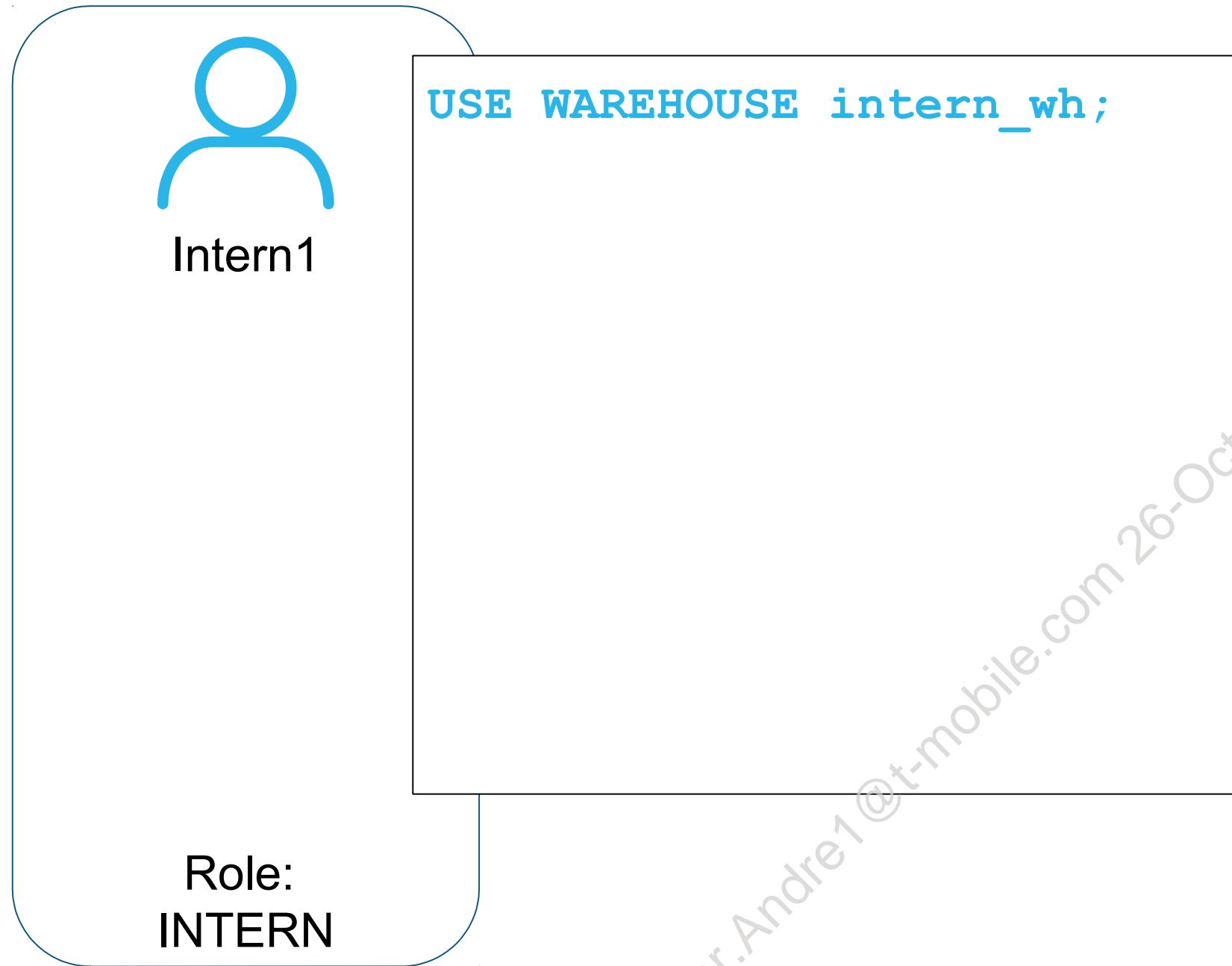


COMPANYA SCENARIO



The user intern1's default role is set to PUBLIC but intern1 has been granted to role INTERN.

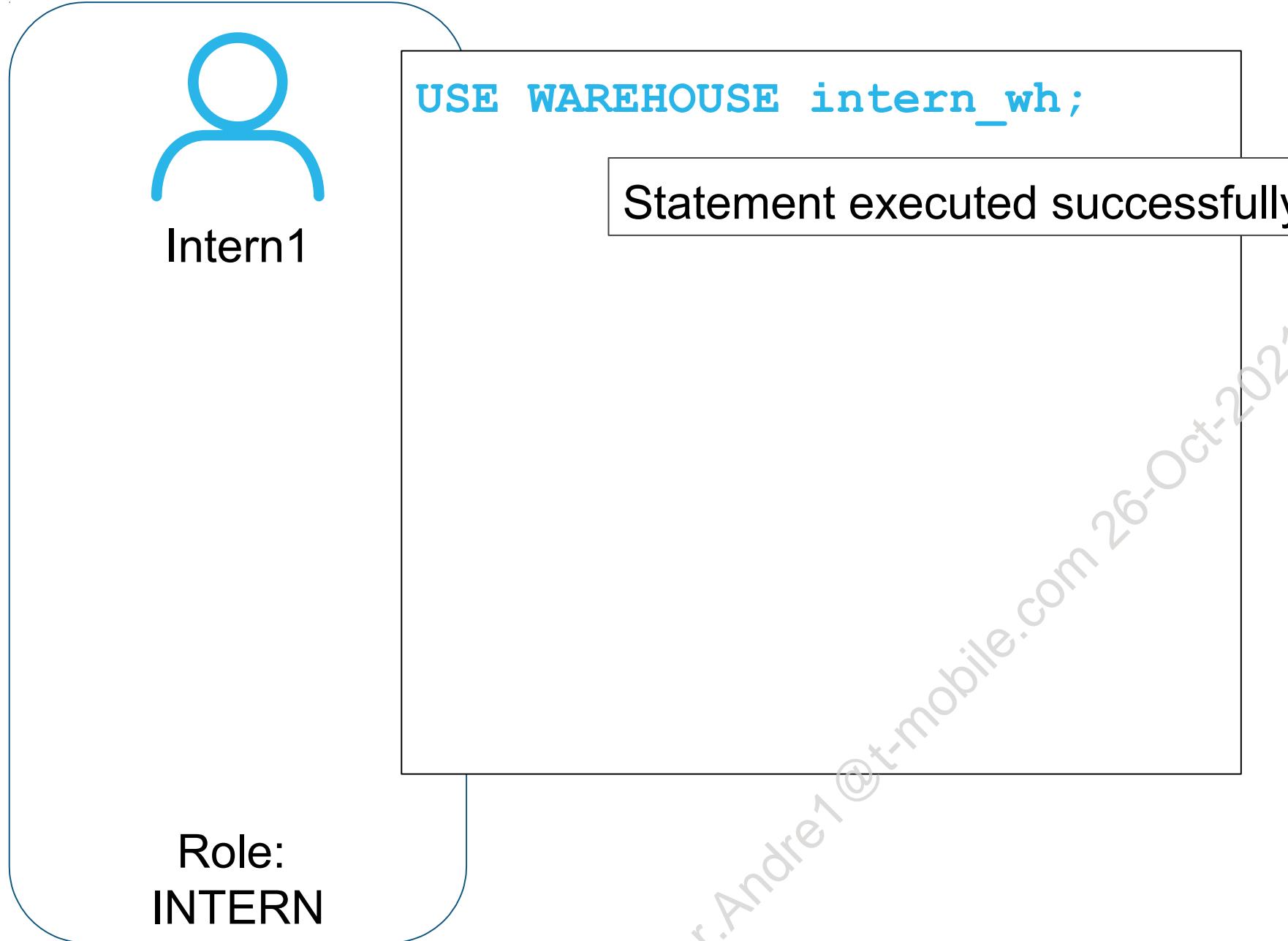
COMPANYA SCENARIO



Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

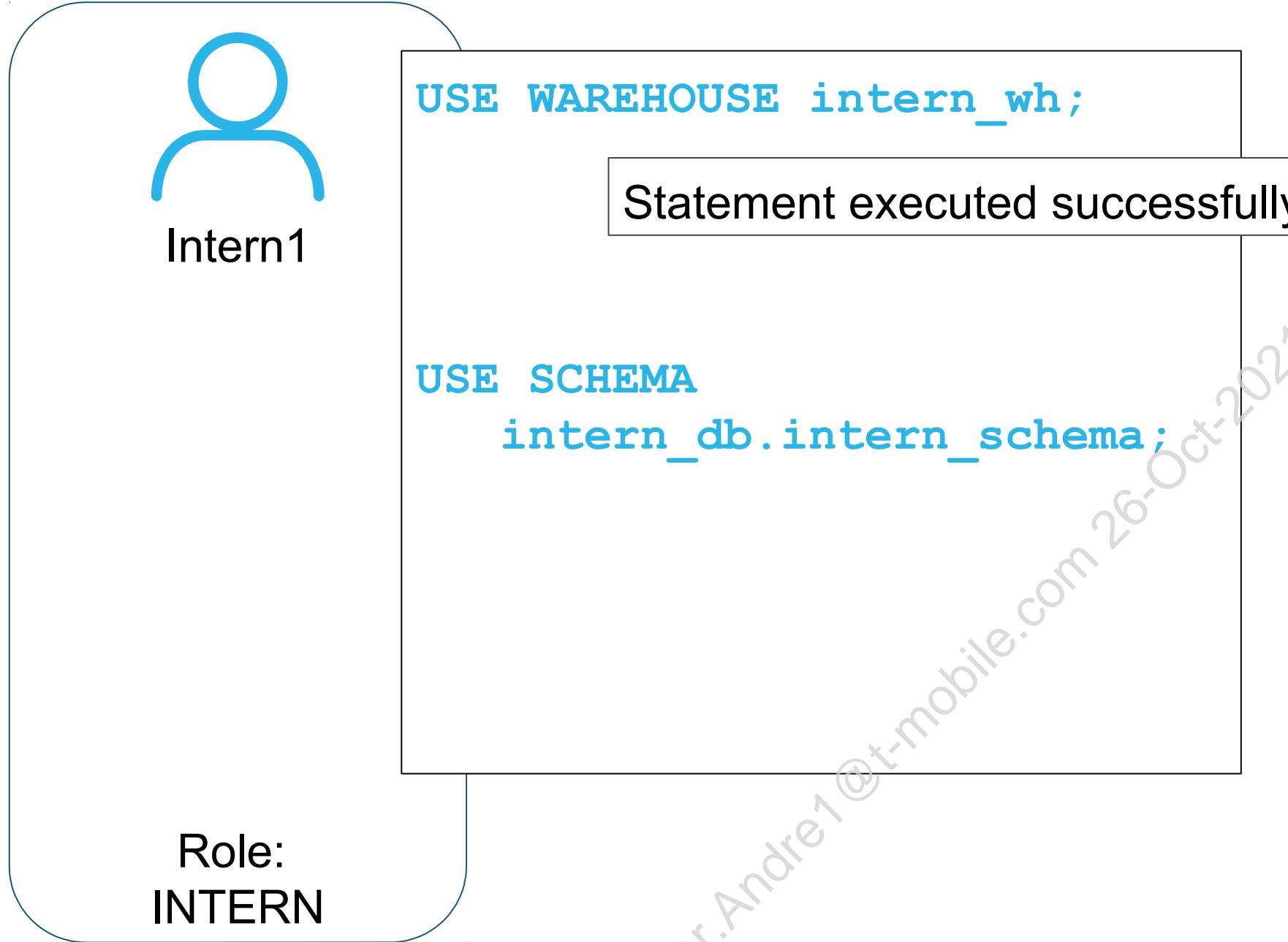


COMPANYA SCENARIO



Using the INTERN role, intern1 can access the `intern_wh` warehouse.
NOTE: Access is dependent on the current role and not the user.

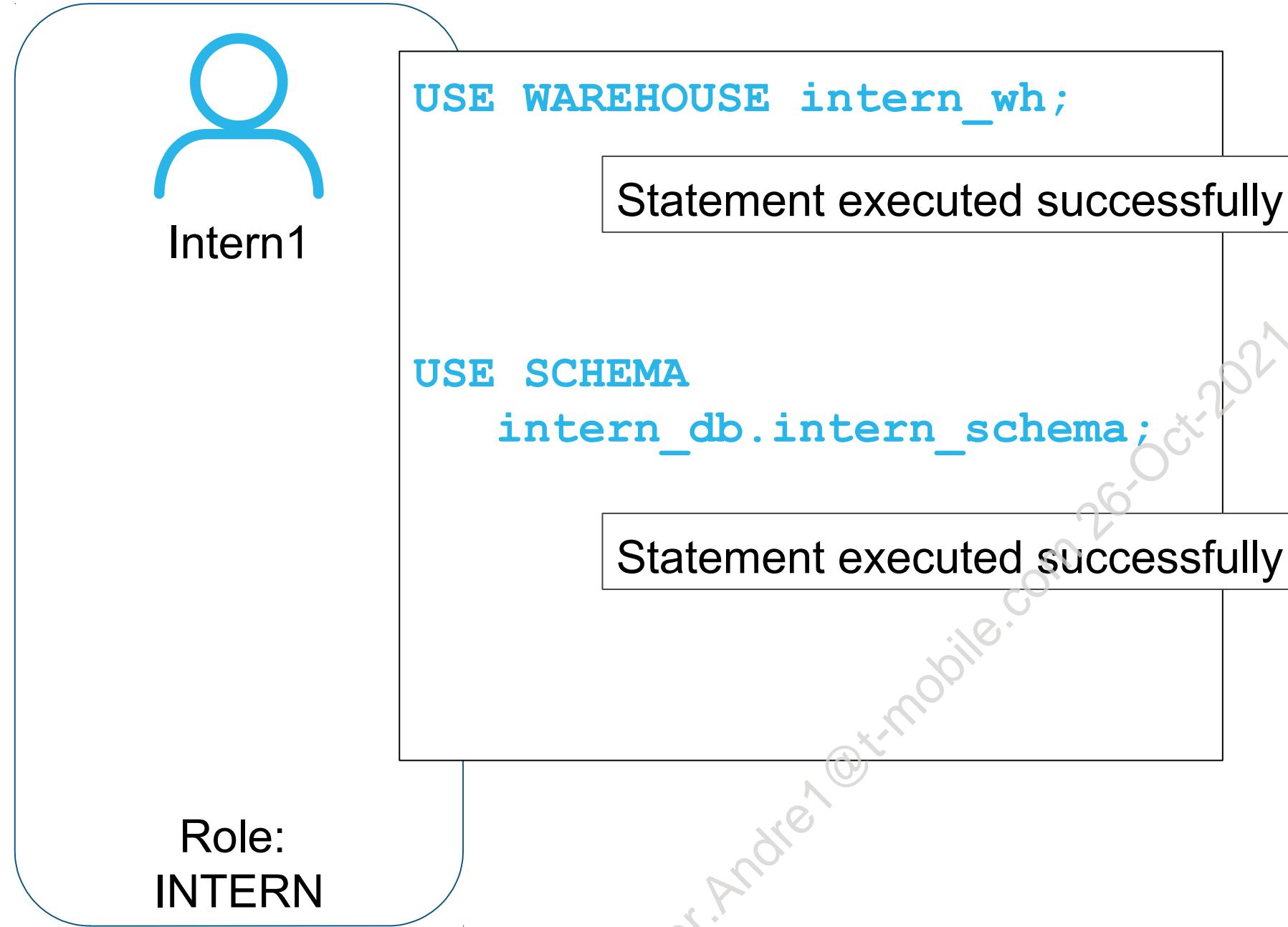
COMPANYA SCENARIO



Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

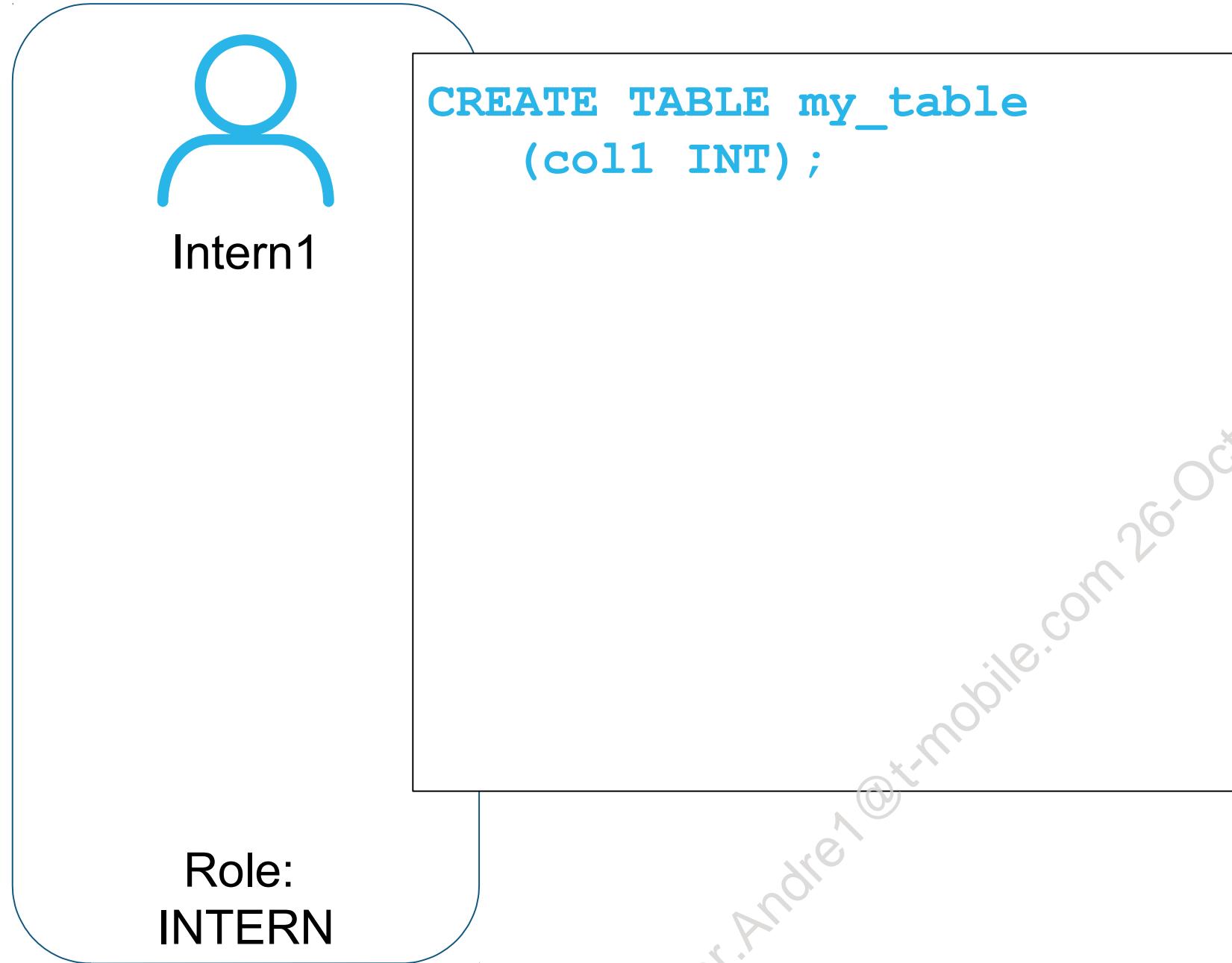


COMPANYA SCENARIO



The INTERN role was granted USAGE permission on the intern_db database and was used to create intern_schema. That means the INTERN role is the owner of the schema.

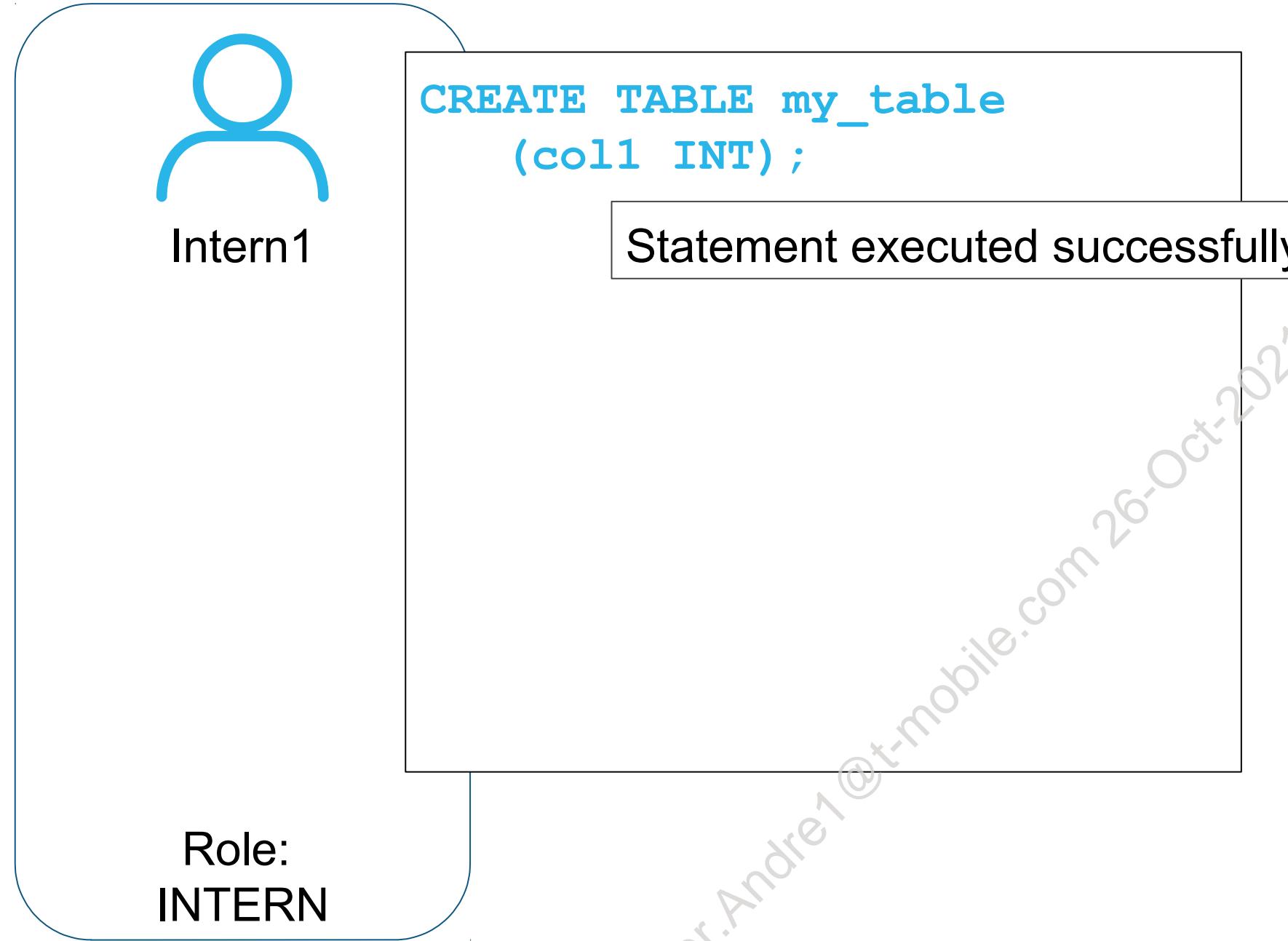
COMPANYA SCENARIO



Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



COMPANYA SCENARIO



The role INTERN has been **granted privileges** to:

- Use the database INTERN_DB
- Use the warehouse INTERN_WH
- Create schemas on database INTERN_DB

Through **ownership**, the role INTERN can:

- Create objects inside the created schemas (which makes INTERN the owner of those objects)
- Do anything with the created objects

LAB EXERCISE: 7

Access Control and User Management

40 minutes

Exercise: Manage Roles

Tasks:

- Creating Parent & Child Roles
- Granting Privileges on Roles
- Dropping Roles



Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

SEMI-STRUCTURED DATA

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Overview
- Query Semi-Structured Data

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



NATIVE SUPPORT

- Native support for all your data
 - Data Lakes with machine-generated data
 - Sensor IoT type data
 - Semi-Structured origin formats (JSON, Avro, ORC, Parquet, or XML)
- Statistics are gathered and maintained, including subfields for quicker query access and effective partition pruning for performance
- Support for all SQL operations (joins, group by, order by, ...)
 - Native syntax for accessing data even in semi-structured fields



SEMI-STRUCTURED DATA TYPES

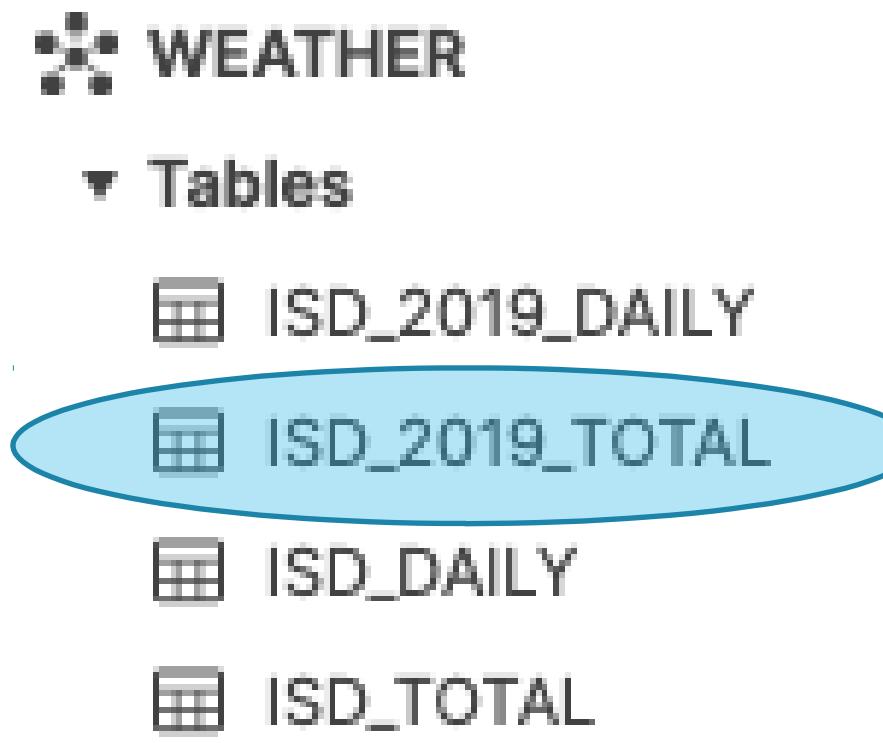


- VARIANT – holds values of standard SQL type, arrays, and objects
 - Non-native values (e.g., dates and timestamps) are stored as strings in a variant column
 - Operations could be slower than when stored in a relational column as the corresponding data type
- OBJECT – a collection of key-value pairs
 - The value is a VARIANT
- ARRAY – Arrays of varying sizes
 - The value is a VARIANT



VARIANT DATA TYPE EXAMPLE

SAMPLE WEATHER DATA SET IN TRAINING_DB



```
{  
  "data": {  
    "observations": [  
      {  
        "air": {  
          "dew-point": 10.6,  
          "dew-point-quality-code": "1",  
          "temp": 11.5,  
          "temp-quality-code": "1"  
        },  
        "atmospheric": {  
          "pressure": 10180,  
          "pressure-quality-code": "1"  
        },  
        "dt": "2019-02-28T06:00:00",  
        "sky": {  
          "ceiling": 450,  
          "ceiling-quality-code": "1"  
        },  
        "visibility": {  
          "distance": 8000,  
          "distance-quality-code": "1"  
        },  
        "wind": {  
          "direction-angle": 140,  
          "direction-quality-code": "1",  
          "speed-quality-code": "1",  
          "speed-rate": 10  
        }  
      }  
    ]  
  }  
}
```



SEMI-STRUCTURED FILE FORMATS

```
[{"table1": {"dim": {"H": 30, "W": 38, "L": 65}, "options": ["teak", "oak", "walnut"]}, {"table2": ...}]}]
```

JSON File

SEMI-STRUCTURED FILE FORMATS

```
[{"table1": {"dim": {"H": 30, "W": 38, "L": 65}}, {"options": ["teak", "oak", "walnut"]}, {"table2": ...} ]
```

The diagram illustrates the structure of a JSON file. It features a large blue bracket on the right side enclosing the entire code block. Inside this bracket, two smaller blue brackets highlight specific parts of the object. The top bracket covers the inner object under the key "table1". The bottom bracket covers the array under the key "options". To the right of these highlighted sections, labels provide definitions: "key" is shown next to the "dim" key under "table1", and "value (object)" is shown next to the "H", "W", and "L" values under "dim". Similarly, "key" is shown next to the "options" key, and "value (array)" is shown next to the three string values "teak", "oak", and "walnut".



SEMI-STRUCTURED FILE FORMATS

```
[{"table1": {"dim": {"H": 30, "W": 38, "L": 65}}, {"options": ["teak", "oak", "walnut"]}, {"table2": ...} ]
```

The diagram illustrates the structure of a JSON file. It features a large blue bracket on the right labeled "JSON File". Inside, there are two main objects. The first object has a key "table1" with a value that is another object ("dim"). This "dim" object contains keys "H", "W", and "L" with values 30, 38, and 65 respectively. The second object has a key "options" with a value that is an array containing three strings: "teak", "oak", and "walnut". Labels "key" and "value (object)" are placed near the top level, while "value (array)" is placed near the bottom level of the array structure.

Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SEMI-STRUCTURED FILE FORMATS

```
[{"table1": {"dim": {"K": "H": 30, "V": 1}, "K": "W": 38, "V": 2}, "K": "L": 65, "V": 3}, {"options": ["teak", "oak", "walnut"]}], {"table2": ...} ]
```

The diagram illustrates the structure of a JSON file. It shows nested objects and arrays. Labels indicate the types of elements:

- key**: Refers to object keys like "table1", "dim", "K", "V", "options", "teak", "oak", and "walnut".
- value (object)**: Refers to object values like {"dim": {"K": "H": 30, "V": 1}, "K": "W": 38, "V": 2}, {"K": "L": 65, "V": 3}, and [{"table2": ...}].
- value (array)**: Refers to array values like ["teak", "oak", "walnut"].

A large blue bracket on the right side of the diagram is labeled **JSON File**.



Roger.Andre1@t-mobile.com 26-Oct-2021 & Snowflake 2020-do-not-copy

SEMI-STRUCTURED FILE FORMATS

Other semi-structured file formats can be loaded into a VARIANT column and queried, using similar commands and functions as JSON data.

- Avro: Open-source framework originally developed for use with Apache Hadoop
- ORC (Optimized Row Columnar): binary format used to store Hive data
- Parquet: binary format designed for projects in the Hadoop ecosystem
- XML (Extensible Markup Language): consists primarily of tags <> and elements



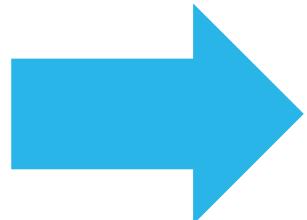
QUERY SEMI-STRUCTURED DATA

Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy



QUERY RICH HIERARCHICAL STRUCTURES IN SQL

{ "data": { "observations": [{ "air": { "dew-point": 1.9, "dew-point-quality-code": "1"...,
[{ "air": { "dew-point": 1.9, "dew-point-quality-code": "1", "temp": 10.2, "temp-quality-code": "1" }, "atmos...
[{ "air": { "dew-point": 3, "dew-point-quality-code": "1", "temp": 14.1, "temp-quality-code": "1" }, "atmos...
[{ "air": { "dew-point": 24.6, "dew-point-quality-code": "1", "temp": 24.6, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": 22.4, "dew-point-quality-code": "1", "temp": 28.5, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": -2.6, "dew-point-quality-code": "1", "temp": 0.8, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": 5.8, "dew-point-quality-code": "1", "temp": 9.6, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": -10.3, "dew-point-quality-code": "1", "temp": -6.5, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": -8.7, "dew-point-quality-code": "1", "temp": 24.1, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": -16.4, "dew-point-quality-code": "1", "temp": -7, "temp-quality-code": "1" }, "atm...
[{ "air": { "dew-point": 20.7, "dew-point-quality-code": "1", "temp": 26.3, "temp-quality-code": "1" }, "atm...



ELEVATION	TEMP	DEW_POINT	WIND_SPEED
42.4	23	16	21
1271.6	-7.8	-10.6	62
1602.6	21	17	0
9.1	12.2	10.6	15
763	-1	-3	21
232	23	16	26
9.1	28	22	31
1790	21	9	0
191.1	26	10	21
27.4	2	-3	57



ACCESSING VALUES IN NESTED DATA

Access a value in a VARIANT column for a particular key

Colon - column:key

```
SELECT v:station.elev  
      AS elevation  
  FROM isd_2019_total
```

...

ELEVATION
=====
763

Brackets - column['key']

```
SELECT v['station']['elev']  
      AS elevation  
  FROM isd_2019_total
```

...

ELEVATION
=====
763



QUERY RICH HIERARCHICAL STRUCTURES IN SQL

```
{"data": { "observations": [ { "air": { "dew-point": 1.9, "dew-point-quality-code": "1"...,  
[ { "air": { "dew-point": 1.9, "dew-point-quality-code": "1", "temp": 10.2, "temp-quality-code": "1" }, "atm...
```

```
SELECT  
    v:station.elev AS elevation,  
    v:data.observations[0].air.temp AS temp,  
    v:data.observations[0].air."dew-point"  
        AS dew_point,  
    v:data.observations[0].wind."speed-rate"  
        AS wind_speed  
FROM  
    weather.isd_2019_total  
LIMIT 10;
```

ELEVATION	TEMP	DEW_POINT	WIND_SPEED
42.4	23	16	21
1271.6	-7.8	-10.6	62
1602.6	21	17	0
9.1	12.2	10.6	15
763	-1	-3	21
232	23	16	26
9.1	28	22	31
1790	21	9	0
191.1	26	10	21
27.4	2	-3	57



DATA FUNCTIONS

Array Creation and Manipulation

ARRAY_AGG
ARRAY_APPEND
ARRAY_CAT
ARRAY_COMPACT
ARRAY_CONSTRUCT
ARRAY_CONSTRUCT_COMPACT
ARRAY_CONTAINS
ARRAY_INSERT
ARRAY_POSITION
ARRAY_PREPEND
ARRAY_SIZE
ARRAY_SLICE
ARRAY_TO_STRING
ARRAYS_OVERLAP

Object Creation and Manipulation

OBJECT_AGG
OBJECT_CONSTRUCT
OBJECT_DELETE
OBJECT_INSERT

JSON and XML Parsing

CHECK_JSON
CHECK_XML
PARSE_JSON
PARSE_XML
STRIP_NULL_VALUE

Extraction

FLATTEN
GET
GET_PATH , :
XMLGET

Conversion

TO_ARRAY
TO_JSON
TO_OBJECT
TO_VARIANT
TO_XML

Casting

AS_<object_type>
AS_ARRAY
AS_BINARY
AS_CHAR , AS_VARCHAR
AS_DATE
AS_DECIMAL , AS_NUMBER
AS_DOUBLE , AS_REAL
AS_INTEGER
AS_OBJECT
AS_TIME
AS_TIMESTAMP_*

Type Checking

IS_<object_type>
IS_ARRAY
IS_BOOLEAN
IS_BINARY
IS_CHAR , IS_VARCHAR
IS_DATE , IS_DATE_VALUE
IS_DECIMAL
IS_DOUBLE , IS_REAL
IS_INTEGER
IS_NULL_VALUE
IS_OBJECT
IS_TIME
IS_TIMESTAMP_*



CASTING DATA FROM VARIANTS

- VARIANTS are often just strings, arrays or numbers
- Without CASTing, they remain VARIANT object types
- Cast them to SQL data types using the :: operator

```
SELECT
    v:station.elev::NUMBER(6,2)
    AS elevation
FROM
    weather.isd_2019_total
LIMIT 5;
```

Row	ELEVATION
1	87.00
2	38.00
3	1735.00
4	549.00
5	11.70



FLATTENING DATA

- VARIANTS may contain nested elements (arrays and objects that contain the data)
- FLATTEN() extracts data from nested elements
- Almost always used with a LATERAL join (to refer to columns from other tables, views, or table function)

```
LATERAL FLATTEN (input=> expression [ options ] )
```

- input => The expression or column that will be unseated into rows
 - The data must be of type VARIANT, OBJECT, or ARRAY



COLUMNS RETURNED BY FLATTEN

SEQ	Unique sequence number associated with input record; not guaranteed to be gap-free or ordered any particular way
KEY	For maps or objects, this column contains the key to the exploded value
PATH	Path to the element within a data structure that needs to be flattened
INDEX	Index of the element, if it is an array; otherwise NULL
VALUE	Value of the element of the flattened array or object
THIS	The element being flattened (useful in recursive flattening)



FLATTENING DATA

```
SELECT * FROM  
TABLE (FLATTEN(input => PARSE_JSON(' [1,2,3]')) );
```

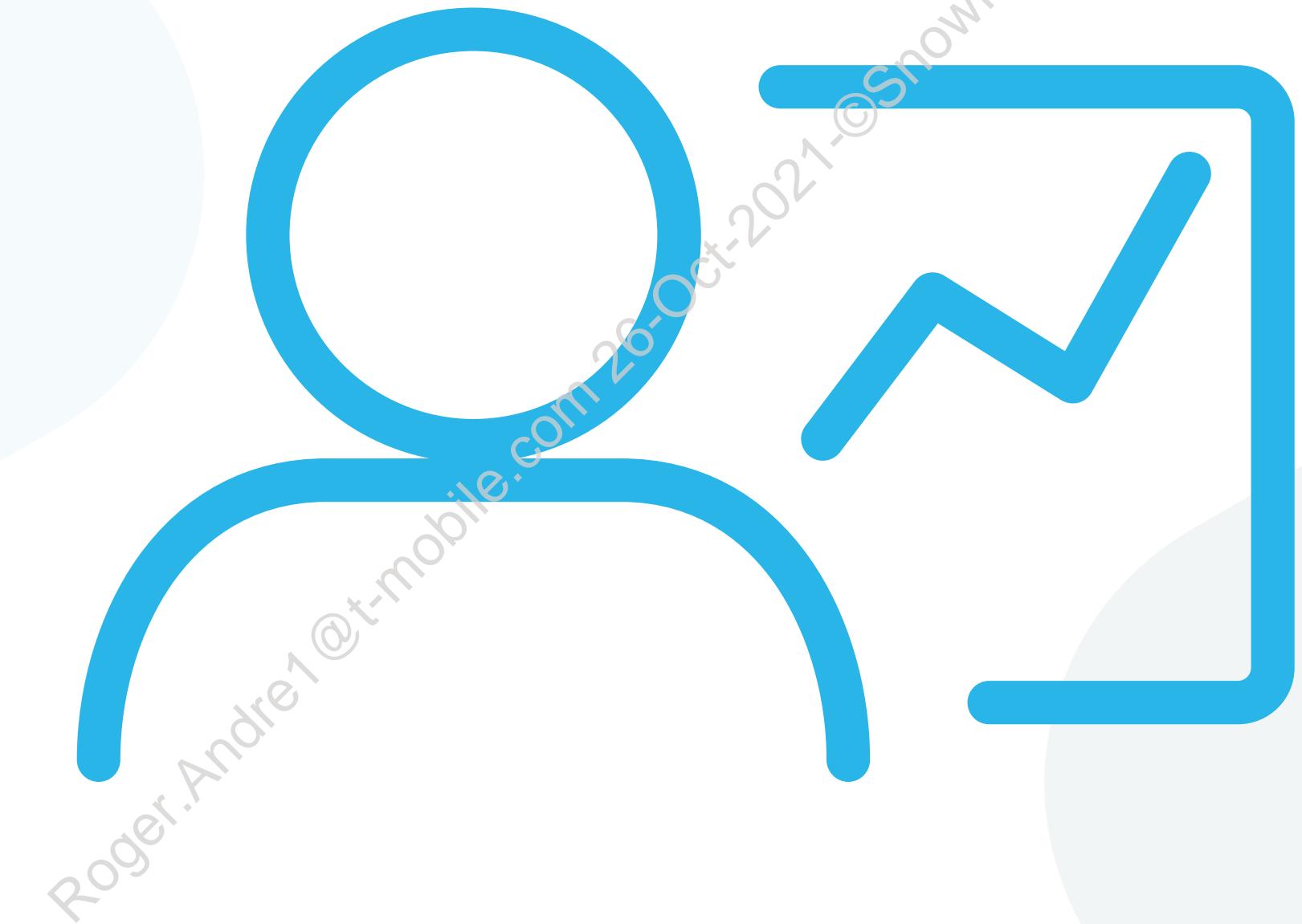
SEQ	KEY	PATH	INDEX	VALUE	THIS
1	NULL	[0]	0	1	[1,2,3]
1	NULL	[1]	1	2	[1,2,3]
1	NULL	[2]	2	3	[1,2,3]



INSTRUCTOR DEMO

View, Query, and Flatten Semi-Structured Data

10 minutes



LAB EXERCISE: 8

Explore Semi-Structured JSON Data

30 minutes

Exercise: Explore Semi-Structured JSON Data

Tasks:

- Review the Weather Data
- Extract & Transform Weather Data

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



CONTINUOUS DATA PROTECTION

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Cloning
- Time Travel
- Failsafe
- Database Replication

Roger.Andre1@t-mobile.com 26-Oct-2021-©Snowflake 2020-do-not-copy

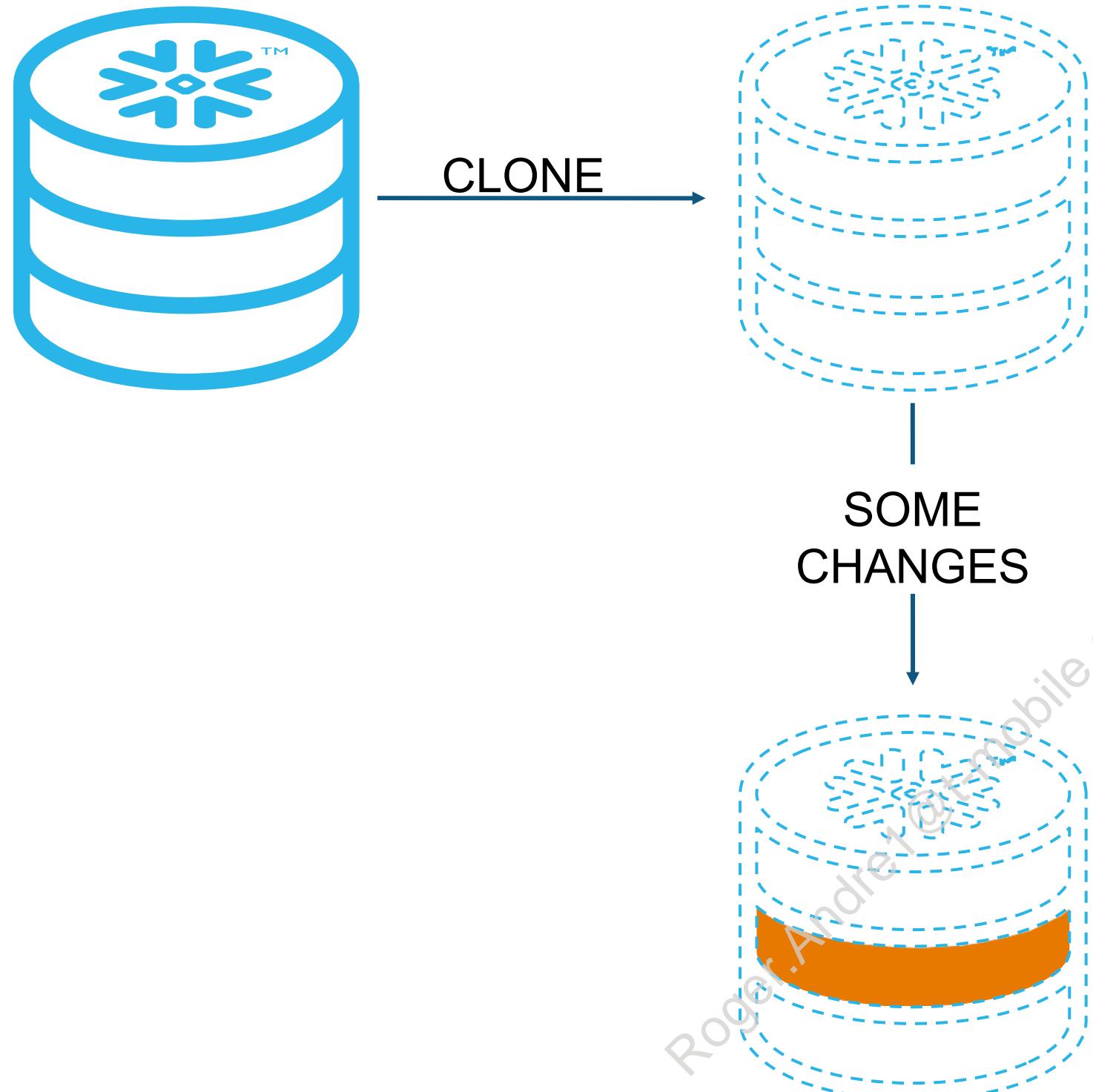


CLONING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



ZERO-COPY CLONING

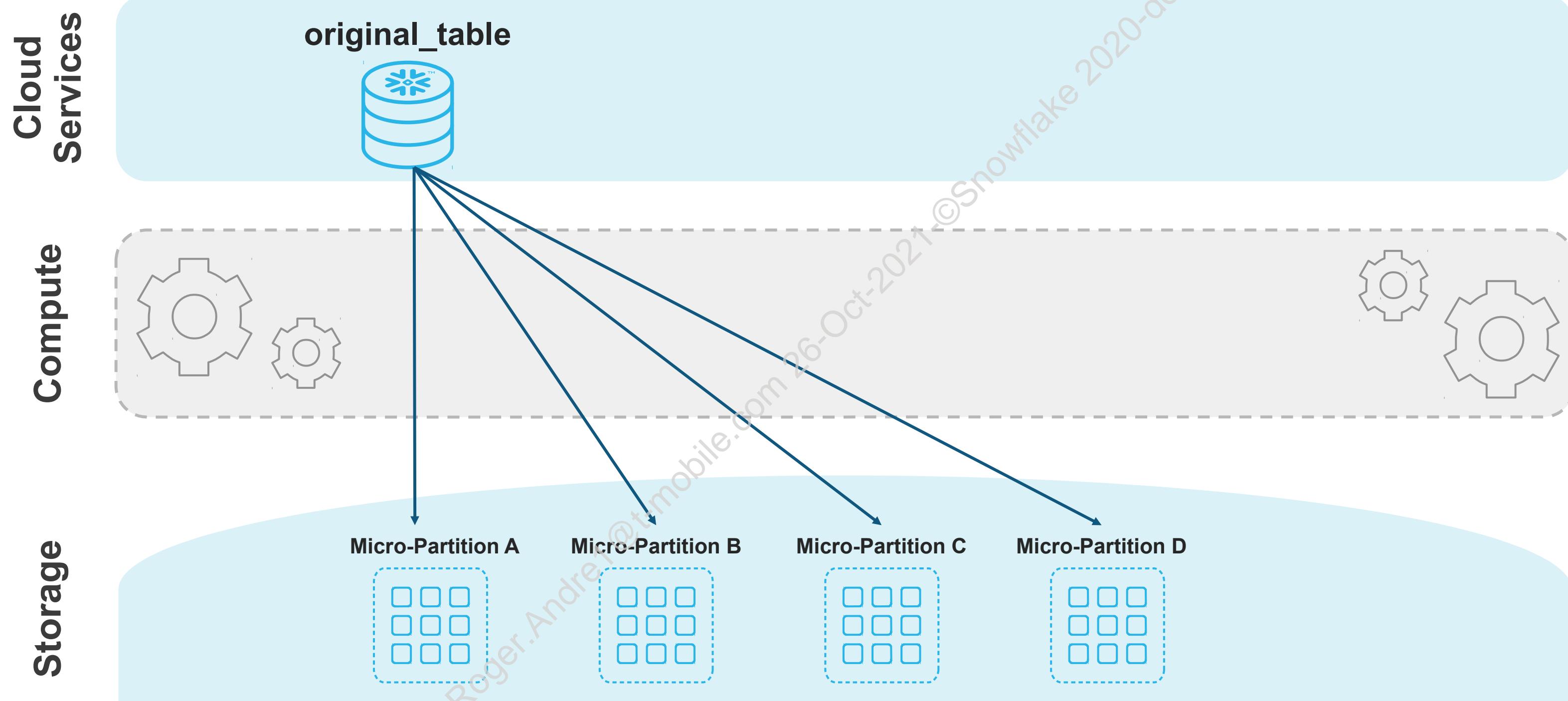


- Quickly take a “snapshot” of any table, schema, or database
- When the clone is created:
 - All micro-partitions in both tables are fully shared
 - Micro-partition storage is owned by the oldest table, clone references them
- No additional storage costs until changes are made to the original or the clone
- Often used to quickly spin up Dev or Test environments
- Effective “backup” option as well



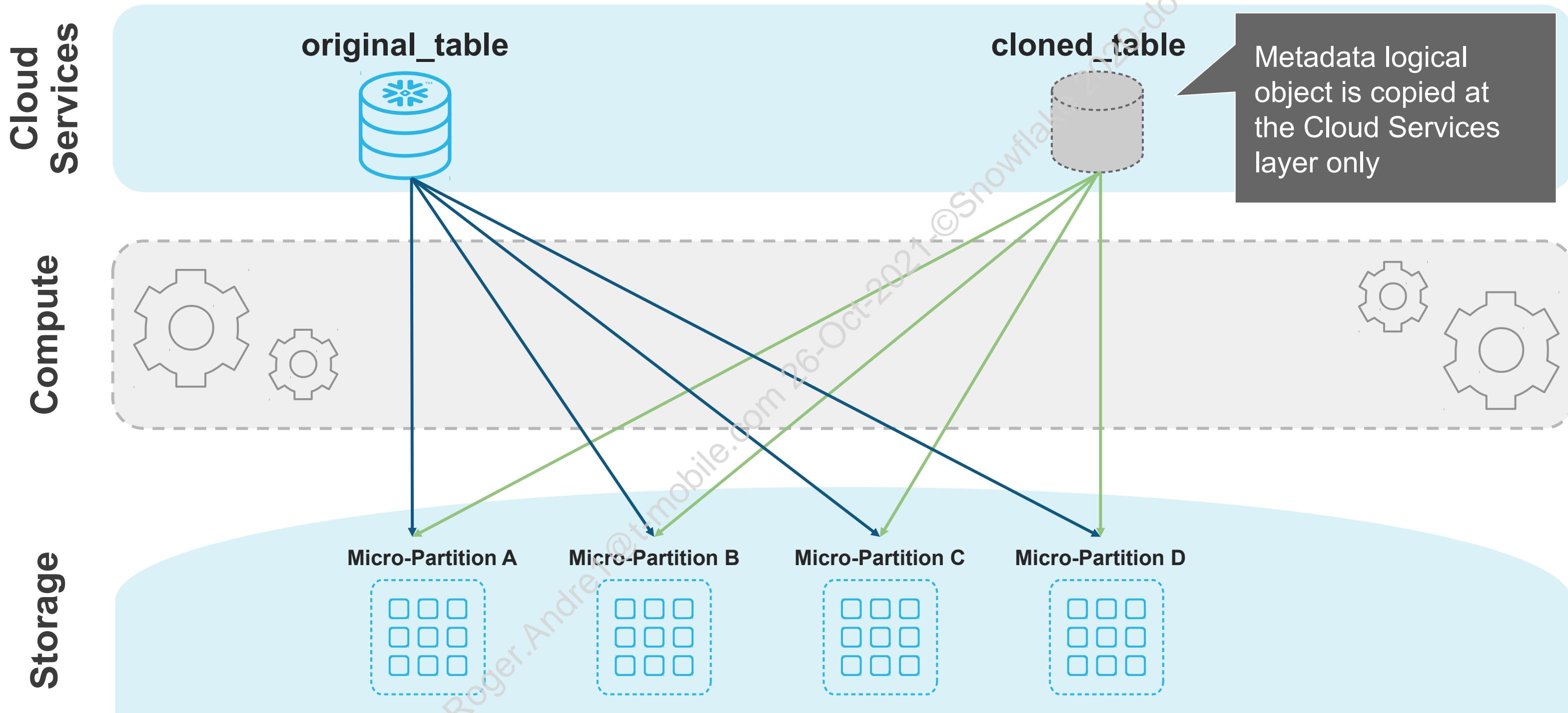
CLONING EXAMPLE

BEFORE CLONE



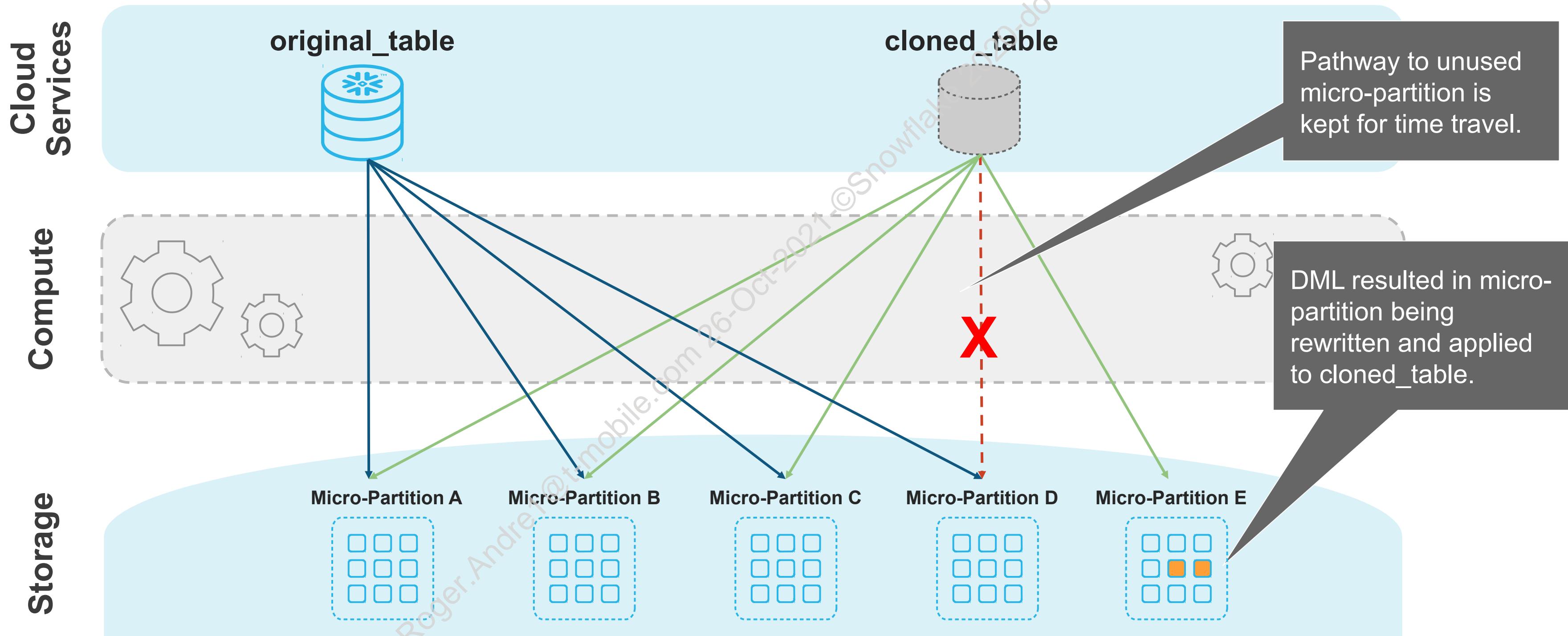
CLONING EXAMPLE

original_table IS CLONED TO cloned_table



CLONING EXAMPLE

original_table IS CLONED TO cloned_table

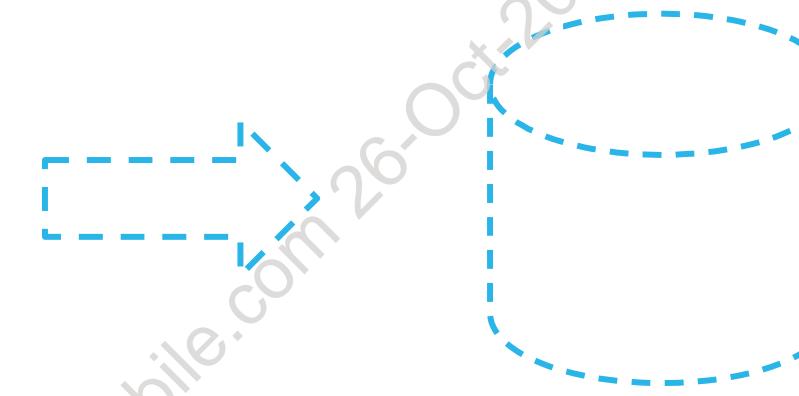


HOW DOES IT WORK?

```
CREATE DATABASE test_db  
CLONE prod_db;
```



PROD_DB



TEST_DB

TIME TRAVEL

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



PROBLEM

RECOVERING FROM MISTAKES



Tables are dropped

Rows are deleted

Schemas are edited

- User errors
- System errors
- Backup itself is a time-consuming task
- Specialized skill and management overhead



SOLUTION: TIME TRAVEL



- Access historical data at any point within a defined retention period
- UNDO common mistakes
- Protect against accidental or intentional modification, removal, or corruption
 - Fix drops, deletes, edits
- Backup/Restore from time or ID

TIME TRAVEL SQL EXTENSIONS



UNDROP <object>

CREATE <object> CLONE... AT | BEFORE

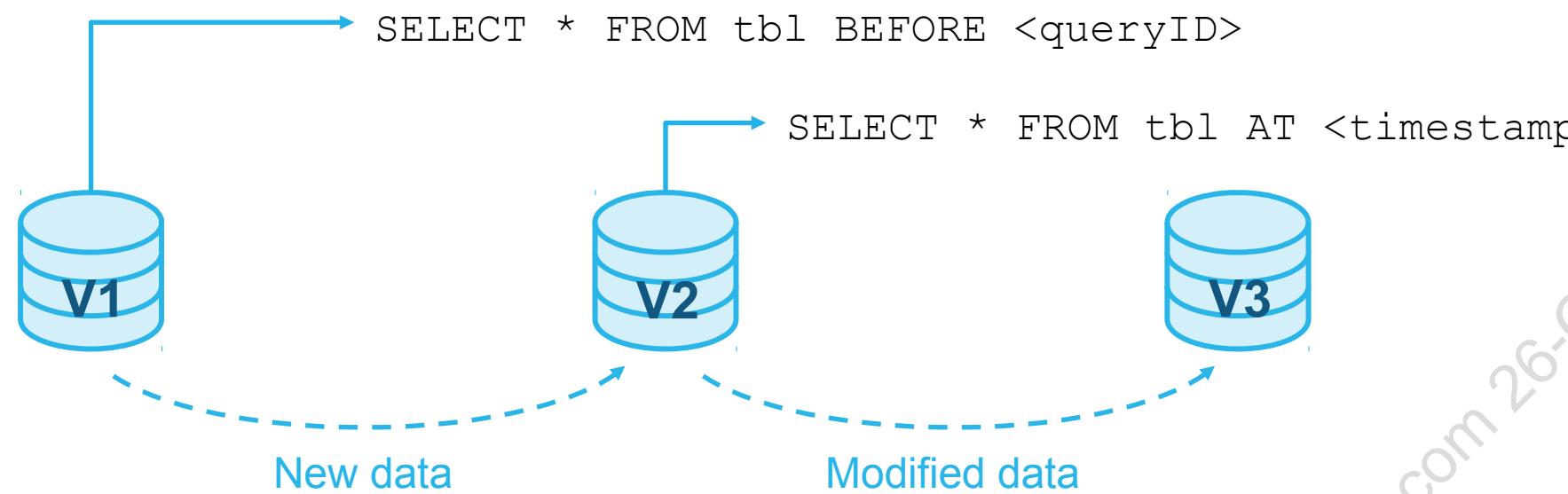
SELECT... FROM <table> AT | BEFORE

DATA_RETENTION_TIME_IN_DAYS

- Instantly bring back deleted tables, schemas, databases
- Restore or duplicate data from key points in the past:
 - Point-in-time
 - Prior to a specific query ID
- Set retention times at the table, schema, database, or account level



TIME TRAVEL



- Available for databases, schemas, and tables
- Configuration retention option:
`DATA_RETENTION_TIME_IN_DAYS`
 - Disable by setting retention to 0
- SQL extensions:
 - `AT | BEFORE` - querying clause
 - `UNDROP` - recovery



CREATE TABLE WITH TIME TRAVEL

- Automatic with default retention period:

```
CREATE TABLE my_table (c1 int);
```

- Customizable retention period:

```
CREATE TABLE my_table (c1 int)  
DATA_RETENTION_TIME_IN_DAYS=90;
```

```
ALTER TABLE my_table  
SET DATA_RETENTION_TIME_IN_DAYS=30;
```



QUERYING WITH TIME TRAVEL

Query Clauses to support Time Travel Actions

- AT a specific time

```
SELECT * FROM my_table1  
AT (TIMESTAMP => 'Mon, 01 May 2020 16:20:00 -0700' ::timestamp);
```

- AT an OFFSET

```
SELECT * FROM my_table1  
AT (OFFSET => -60 * 5);
```

- BEFORE a specific query

```
SELECT * FROM my_table1  
BEFORE (STATEMENT => '8e5d0ca9-005e-44e6-b858-a8f5b37c5726');
```



TIME TRAVEL AND CLONING EXAMPLES

- Cloning Historical Objects

```
CREATE TABLE restored_table CLONE my_table1  
  AT (TIMESTAMP => 'Mon, 09 May 2020 01:01:00 +0300'::timestamp);
```

```
CREATE DATABASE restored_db CLONE my_db  
  BEFORE (STATEMENT => '8e5d0ca9-005e-44e6-b858-a8f5b37c5726');
```

- Restoring Objects

UNDROP TABLE/SCHEMA/DATABASE

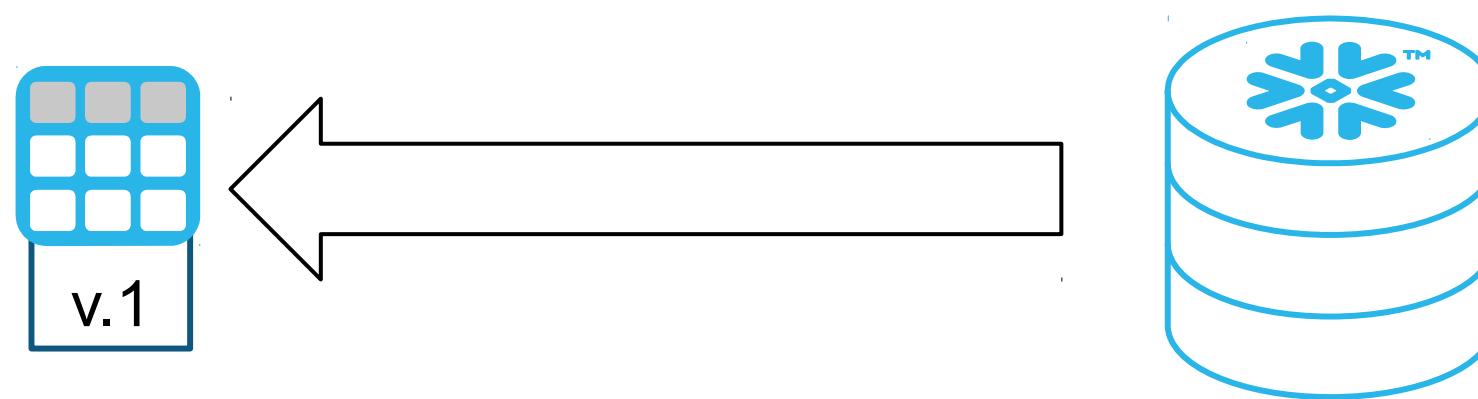


HOW DOES IT WORK?

- Micro-partitions!
- Micro-partitions are immutable
- When data is changed, new versions of the micro-partitions are created
- We keep the older version for the specified retention time



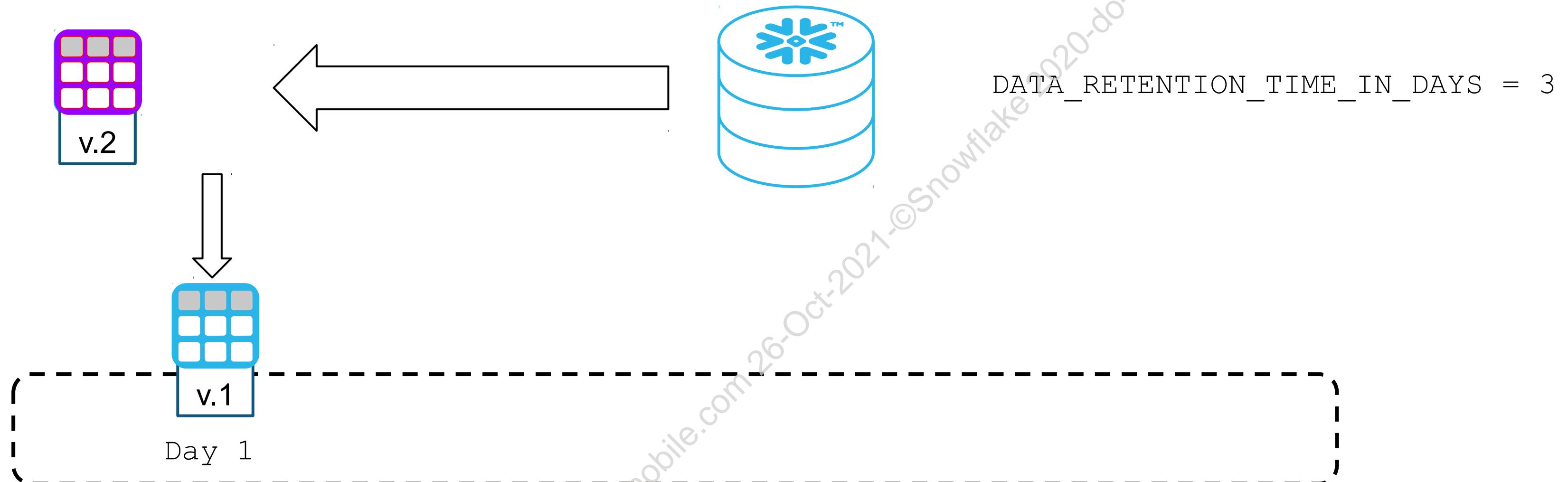
TIME TRAVEL OVERVIEW



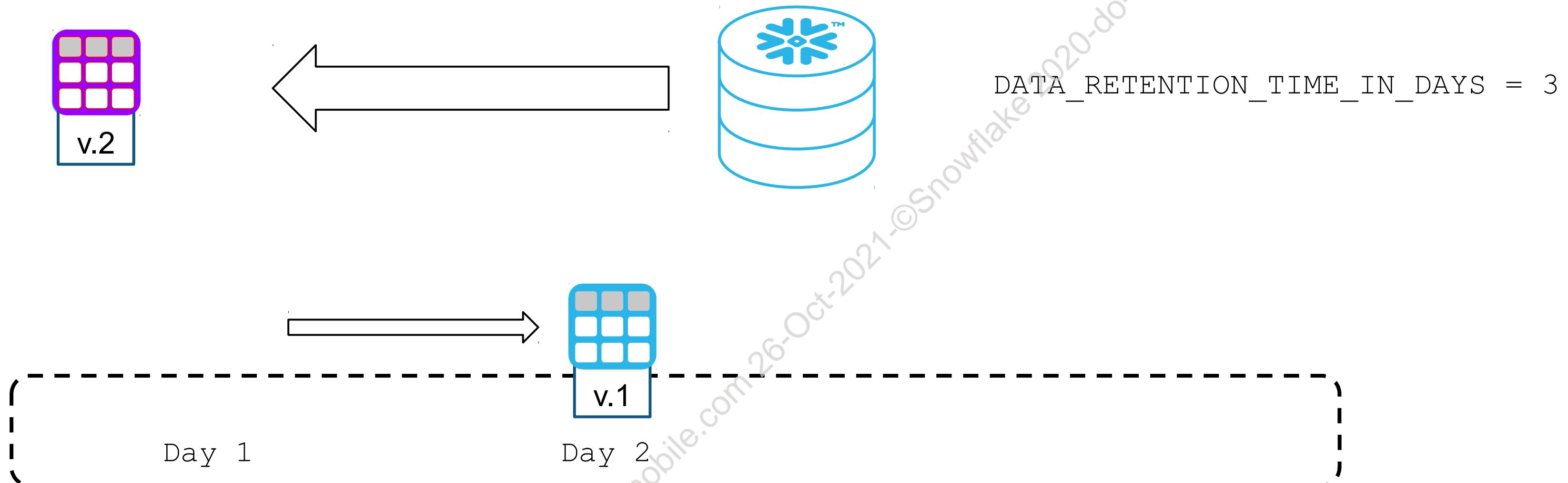
DATA_RETENTION_TIME_IN_DAYS = 3



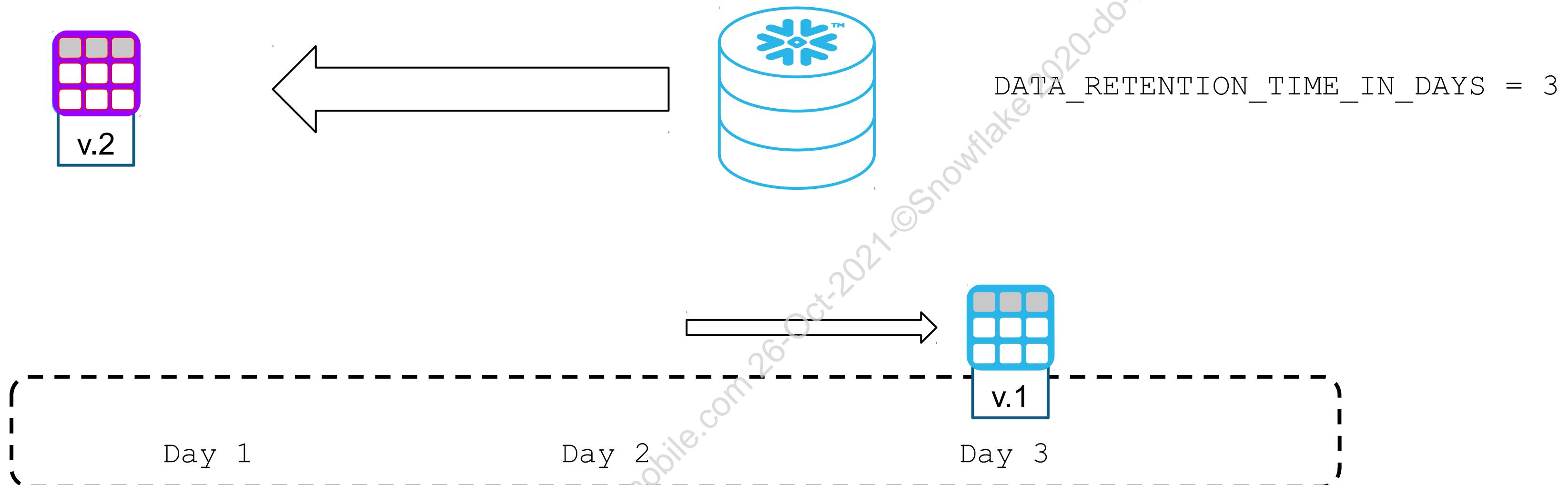
TIME TRAVEL OVERVIEW



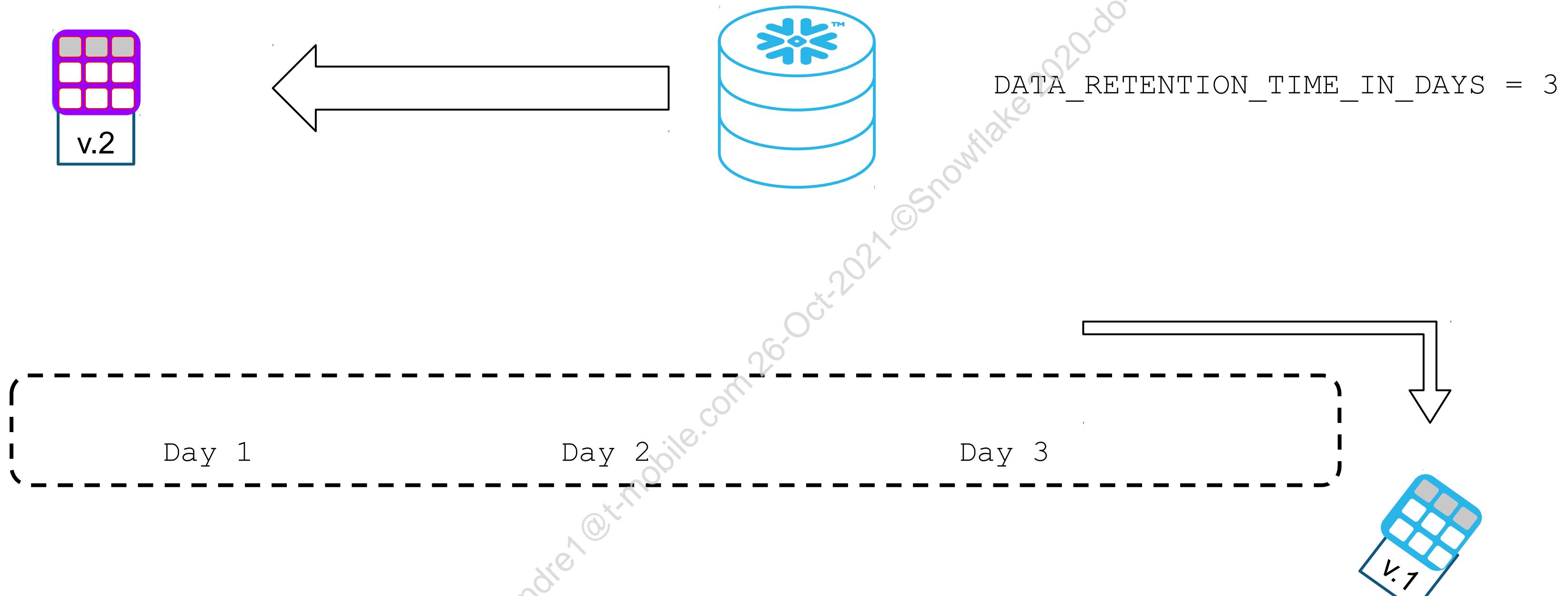
TIME TRAVEL OVERVIEW



TIME TRAVEL OVERVIEW



TIME TRAVEL OVERVIEW

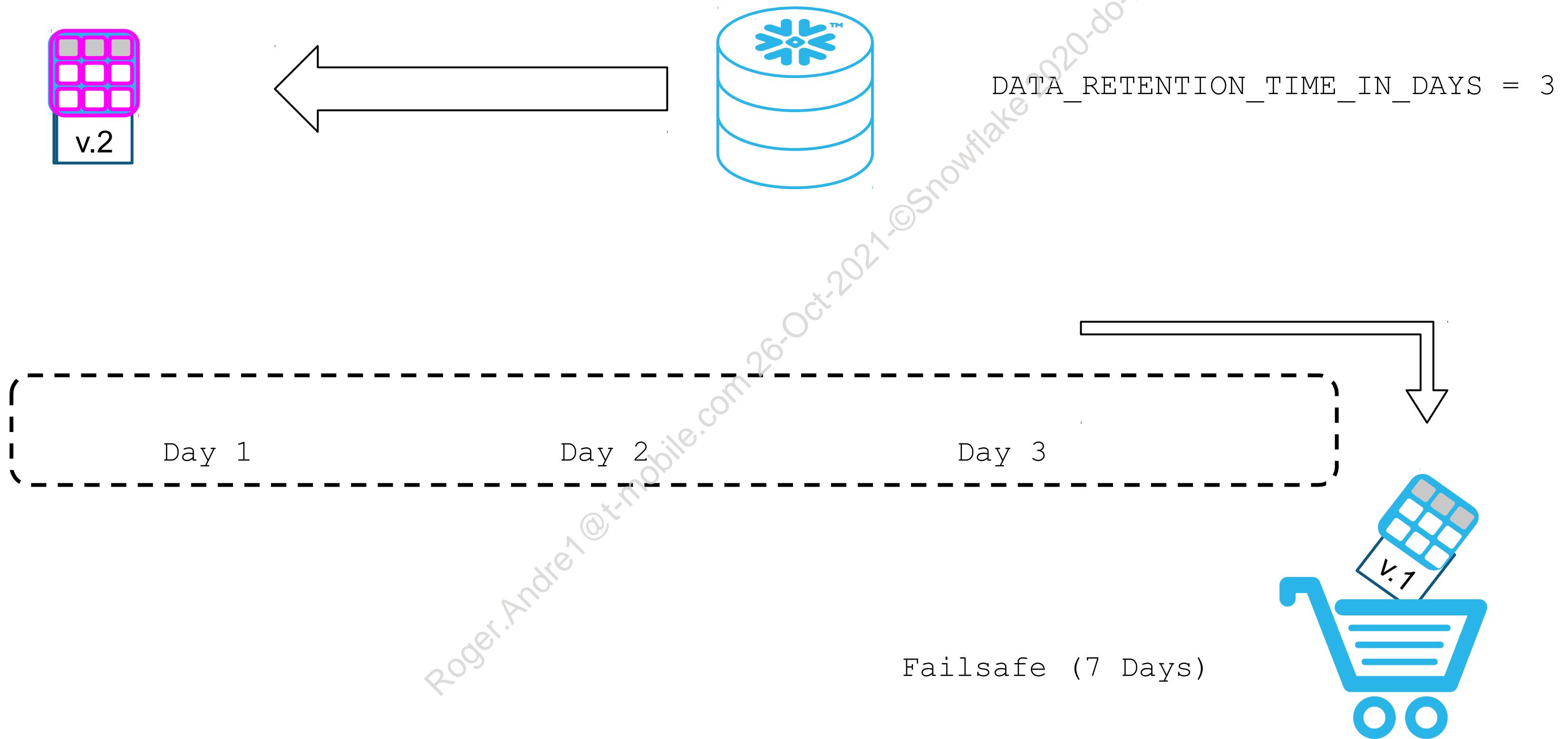


FAILSAFE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



FAILSAFE OVERVIEW



FAILSAFE STORAGE

- Non-configurable, 7-day retention for historical data after Time Travel expiration
- Only accessible by Snowflake personnel
- Admins can view failsafe use in the Snowflake Web UI under *Account > Billing & Usage*
- Not supported for temporary or transient tables



DATA PROTECTION OPTIONS BY EDITION

Snowflake Edition	Time Travel (1 day)	Time Travel (90 days max)	Failsafe (7 days)
Standard	✓		✓
Enterprise		✓	✓
Business Critical		✓	✓
VPS		✓	✓



LAB EXERCISE: 9

Continuous Data Protection

30 minutes

Tasks:

- Clone database objects
- DROP and UNDROP a table
- Recover a table to a time before a change was made
- Object naming constraints

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



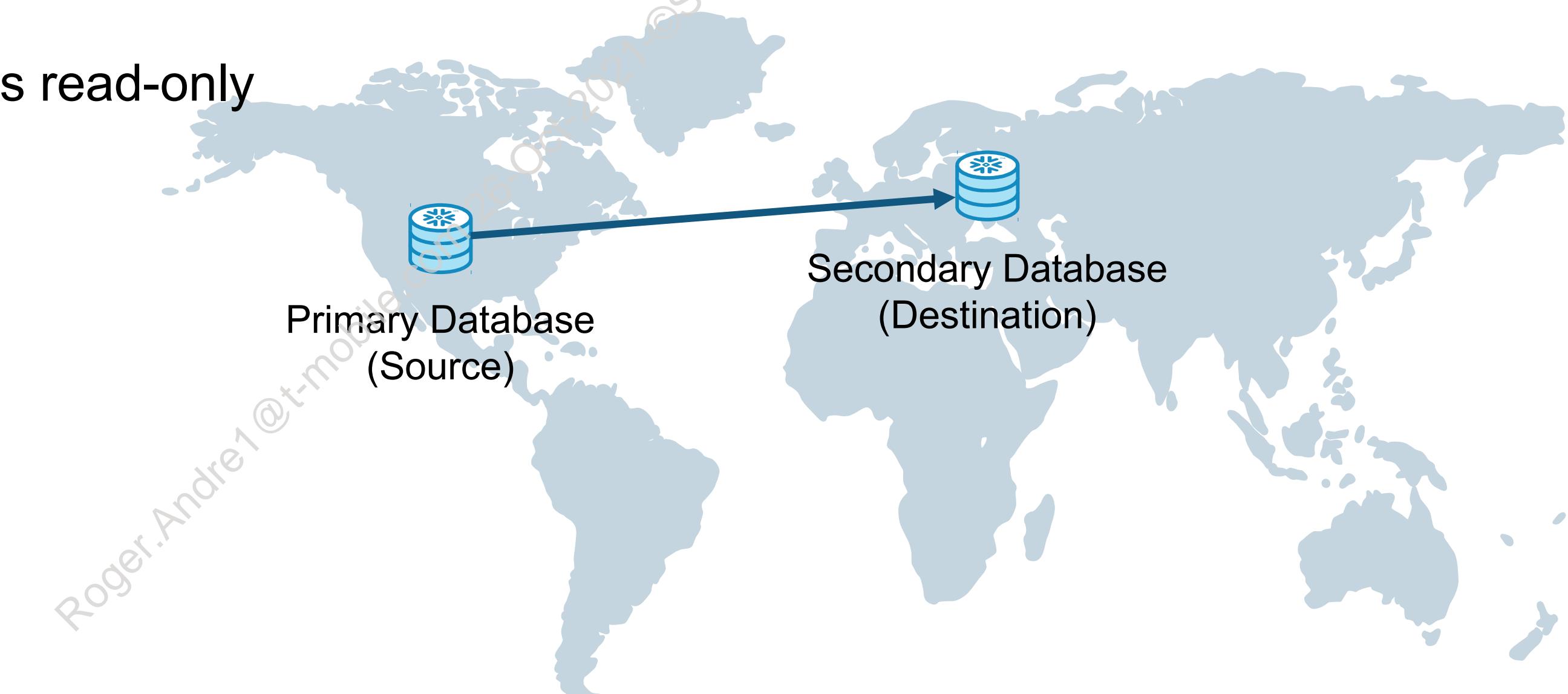
DATABASE REPLICATION

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



WHAT IS REPLICATION?

- Keeps database objects and stored data synchronized between one or more accounts (within the same organization)
- Unit of replication is a database (permanent or transient)
- Secondary database is read-only



REPLICATION USE CASES

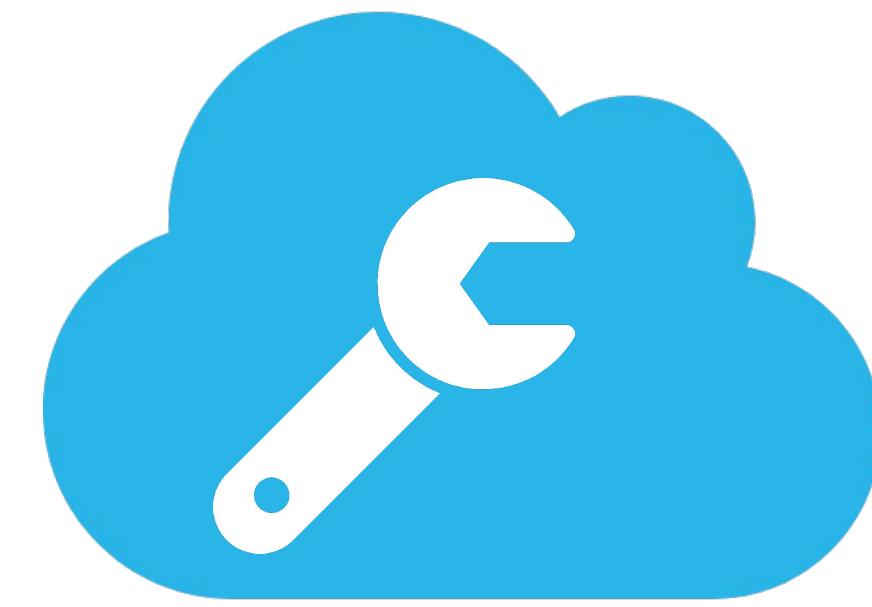
Business Continuity &
Disaster Recovery



Secure Data Sharing
Across Regions / Clouds

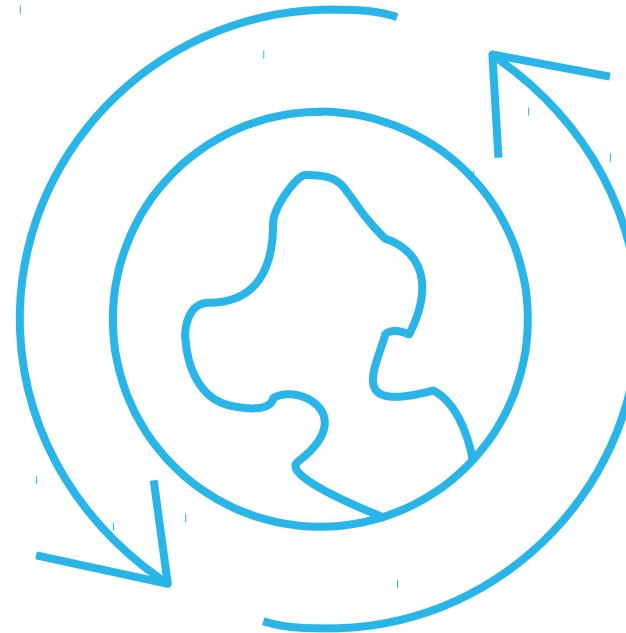


Data Portability for
Account Migrations



REPLICATED DATABASE OBJECTS

Business Continuity &
Disaster Recovery



- Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 Do-not-copy
- Tables
 - Permanent
 - Transient
 - Temporary
 - Clustered Constraints
 - Sequences
 - Views
 - Standard
 - Materialized
 - Stored Procedures
 - User-Defined Functions (UDF)



DATABASE REPLICATION AND ENCRYPTION

- Snowflake encrypts database files in-transit from the source account to the target account
- If Tri-Secret Secure is enabled (for the source and target accounts) the files are encrypted using the public key for an encryption key pair
- The encryption key pair is protected by the account master key (AMK) for the target account



COMPUTE FOR DATABASE REPLICATION

- Replication operations use Snowflake-provided compute resources to copy data between accounts
- Replication utilization is shown as a special Snowflake-provided warehouse named  **REPLICATION**.
- Query either of the following
 - `REPLICATION_USAGE_HISTORY` table function
 - `REPLICATION_USAGE_HISTORY` view



DATA TRANSFER FOR DATABASE REPLICATION

- Initial database replication and subsequent synchronization operations incur data transfer charges
 - These are termed “egress fees” and only apply when data is passed across regions
 - The cost is passed along to customers
- Rate or pricing is determined by the location of the source and target accounts, and the cloud provider
- Data Transfer Usage is shown in **Billing & Usage Data Transfer**



DATA SHARING

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Overview
- Data Providers
- Data Consumers
- Reader Accounts
- The Data Marketplace

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA SHARING

DATA SOURCES



One Platform,
One Copy of Data,
Many Workloads

Secure & Governed
Access to All
Data

Virtually Unlimited
Performance
and Scale

Near-Zero
Maintenance,
as a Service

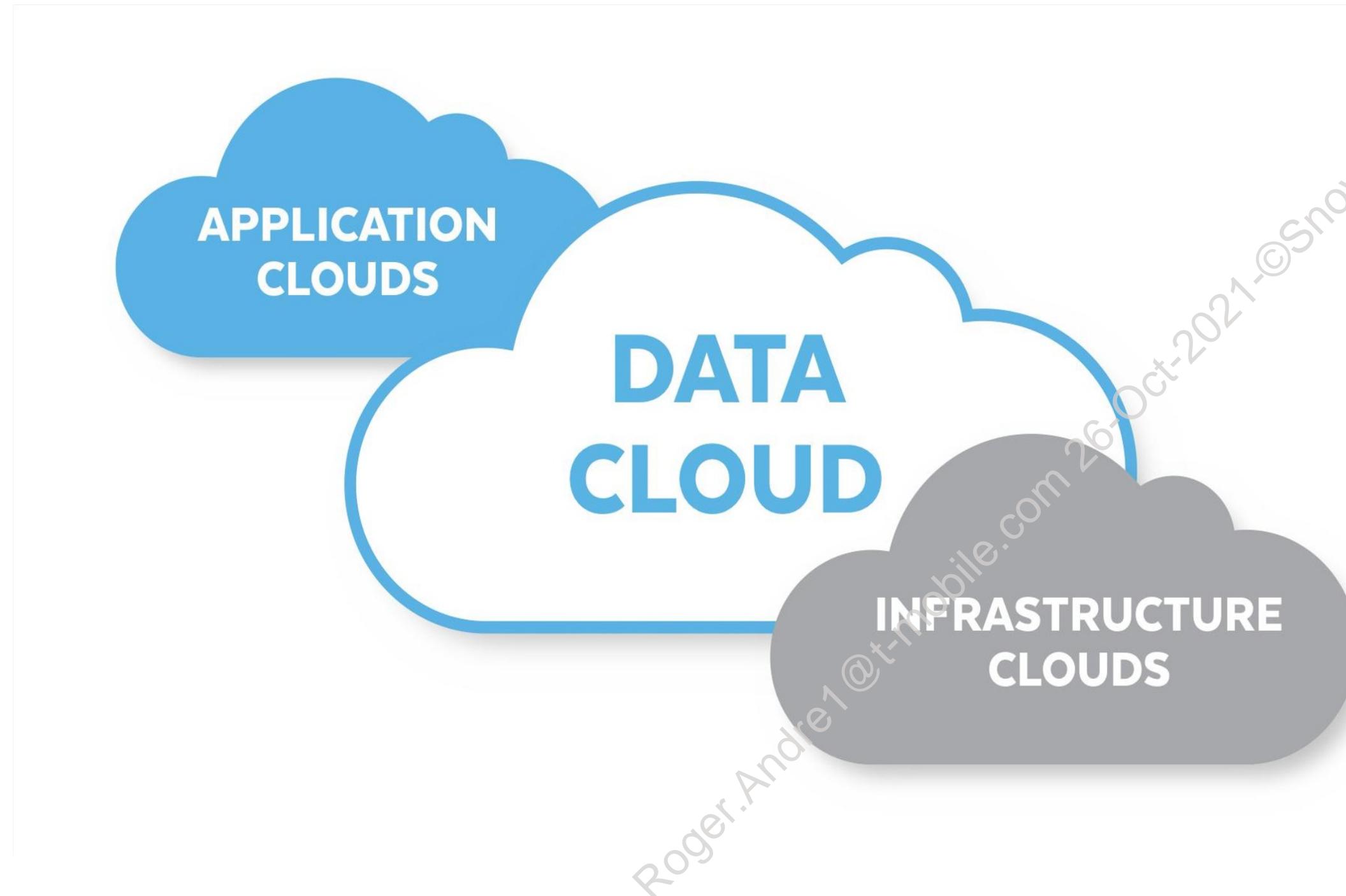


THE DATA CLOUD: A GLOBAL ECOSYSTEM



Roger.Andre1@tmobile.com
Oct-2021-Snowflake.com

WHAT IS THE DATA CLOUD?



THE SOLUTION:

The Data Cloud

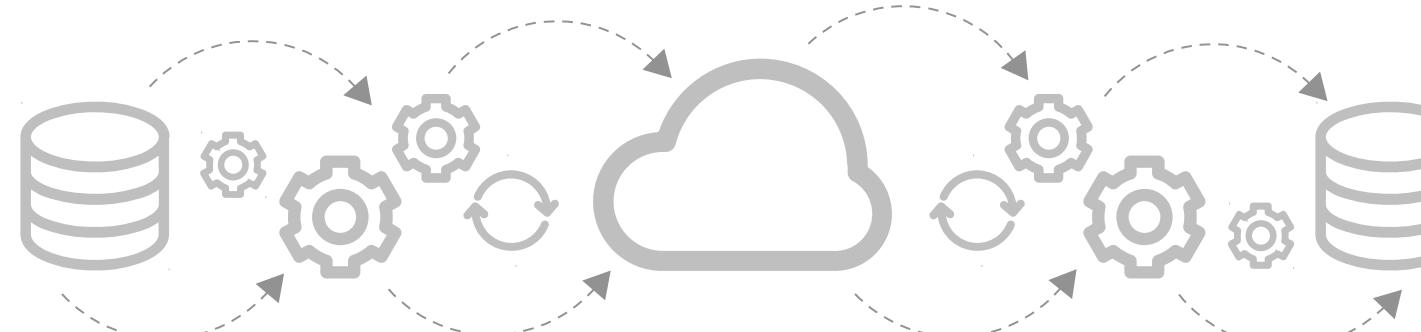
WHAT IS IT?

Snowflake Cloud Data Platform technology plus the data itself.

In the Data Cloud, every Snowflake customer can connect and share data in a completely governed and secure manner

TRADITIONAL WAYS OF SHARING DATA

Traditional Methods



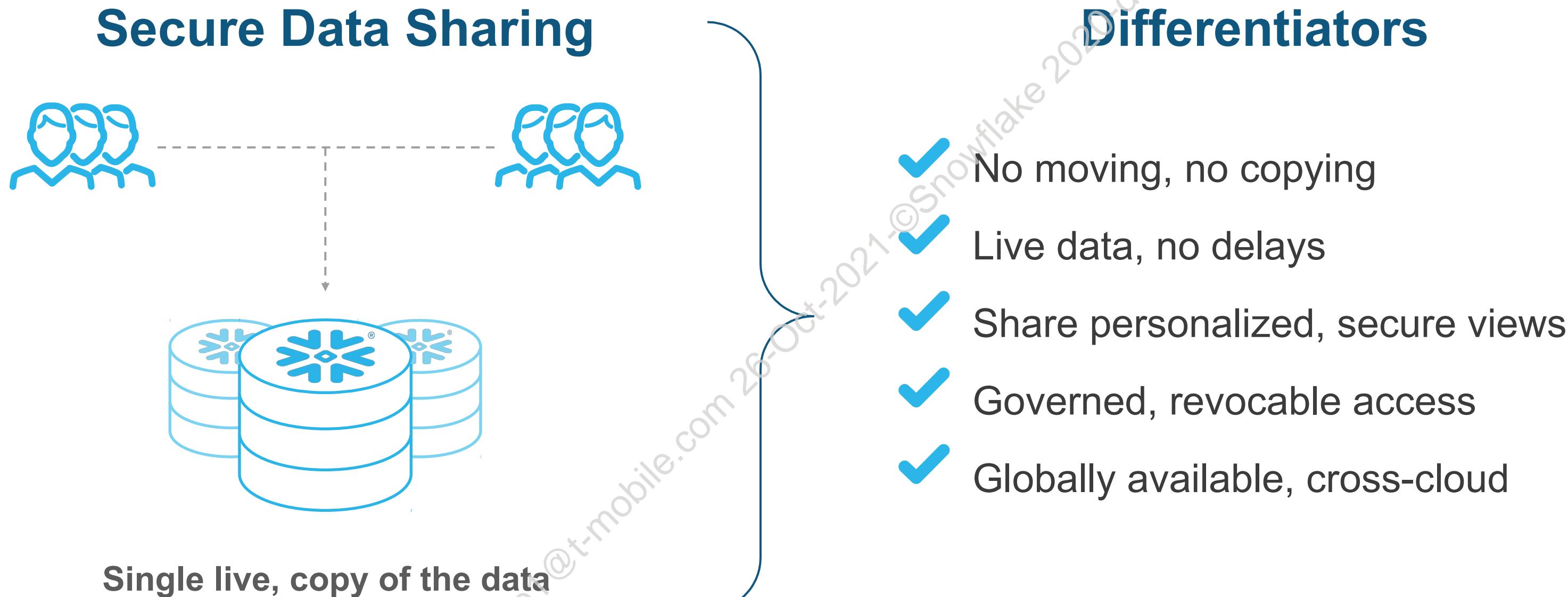
- Copying files in FTP/ cloud buckets
- Building, Maintaining & Calling APIs
- ETL pipelines
- Data marts

Gaps

- ✖ Copy and move data
- ✖ Costly to maintain
- ✖ Data is delayed
- ✖ Error-prone
- ✖ Unsecure, once data is moved



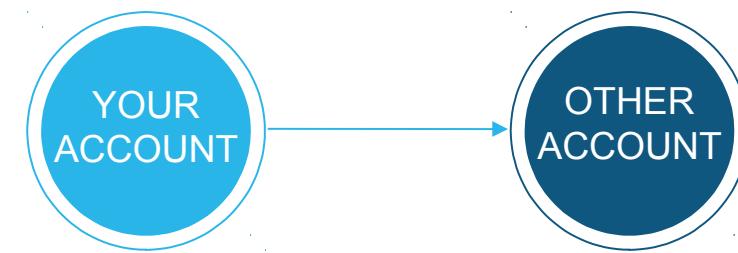
SECURE DIRECT DATA SHARING



PRODUCT OFFERINGS

All powered by Secure Data Sharing: A single, live copy of the data: no copying, no moving, no delays.

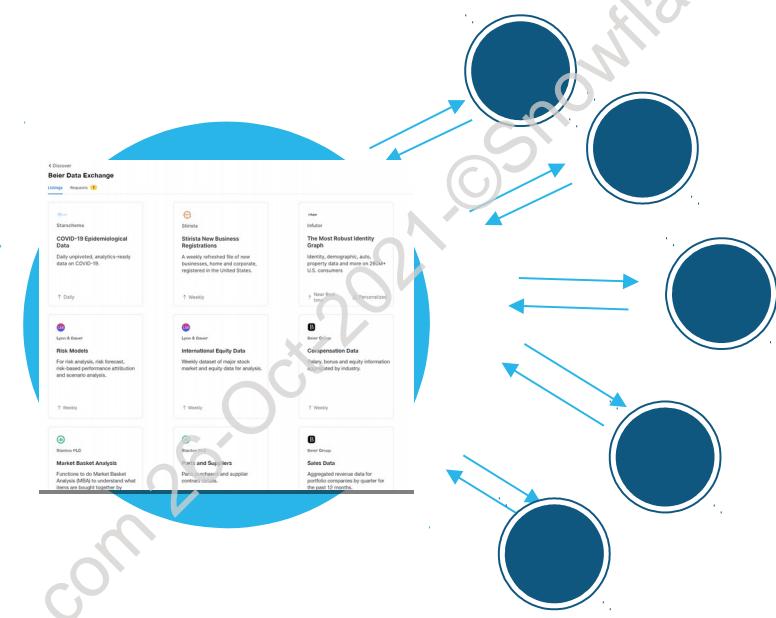
SECURE DIRECT DATA SHARING



Direct Data sharing
1-to-1/many

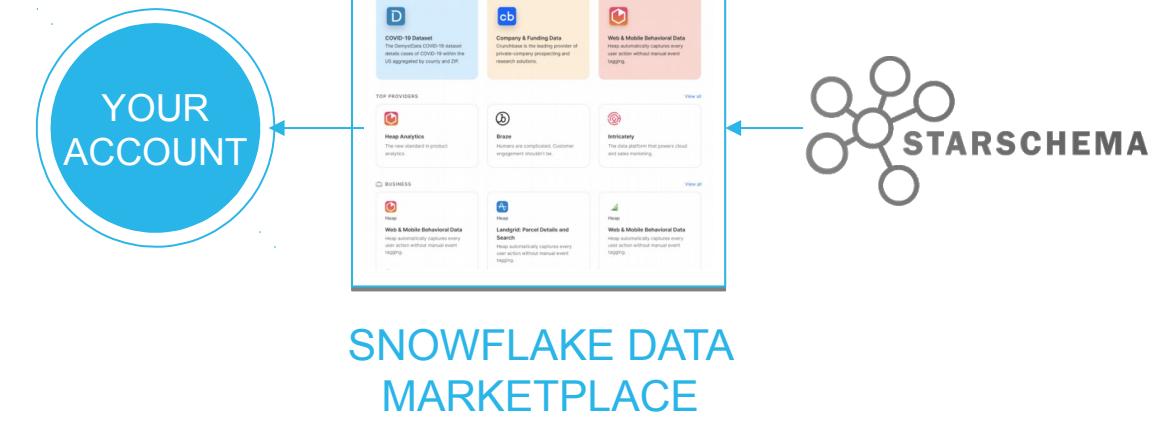
SNOWFLAKE DATA EXCHANGE

YOUR DATA EXCHANGE IN YOUR ACCOUNT



Your Data privately published to your consumers
many parties, many data sets

SNOWFLAKE DATA MARKETPLACE



Publicly publish/consume data from 3rd parties:
Access data from over 60 providers



SNOWFLAKE DATA MARKETPLACE

Snowflake is the only solution for accessing live, personalized data globally

The screenshot shows the Snowflake Data Marketplace interface. It includes sections for 'NEW' products, 'RECENTLY VIEWED' items, and 'TOP PROVIDERS'. The 'NEW' section features three cards: 'Property Assessor' (AI-driven real estate analytics), 'AbiliTec: Resolve Fragmented PII' (connects customer data with LiveRamp's identity resolution platform), and 'Web & Mobile Behavioral Data' (Heap automatically captures user actions). The 'RECENTLY VIEWED' section lists 'AgilOne Customer Data Plat...' and 'The Truth in B2B Intent Data'. The 'TOP PROVIDERS' section lists 'Heap Analytics', 'Braze', and 'Intricately'.

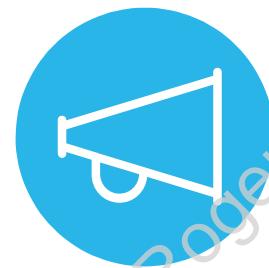
✓ Live, ready-to-query data;
no copying or moving

✓ Only data marketplace
with personalized data

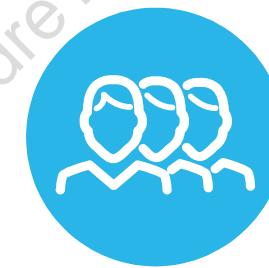
✓ Globally available, across
clouds



Financial



Marketing



Demographic



Macroeconomic



Government



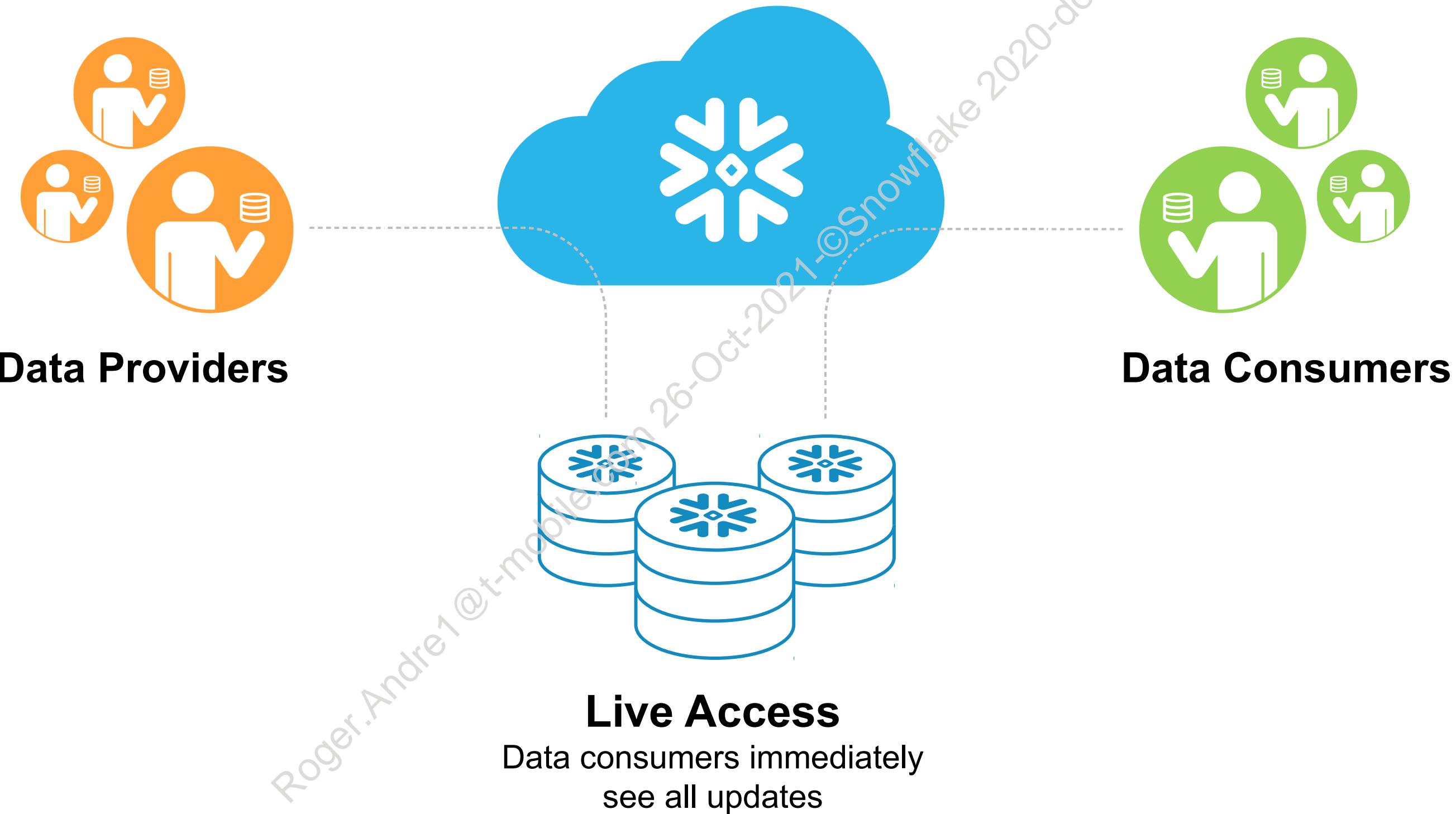
Healthcare



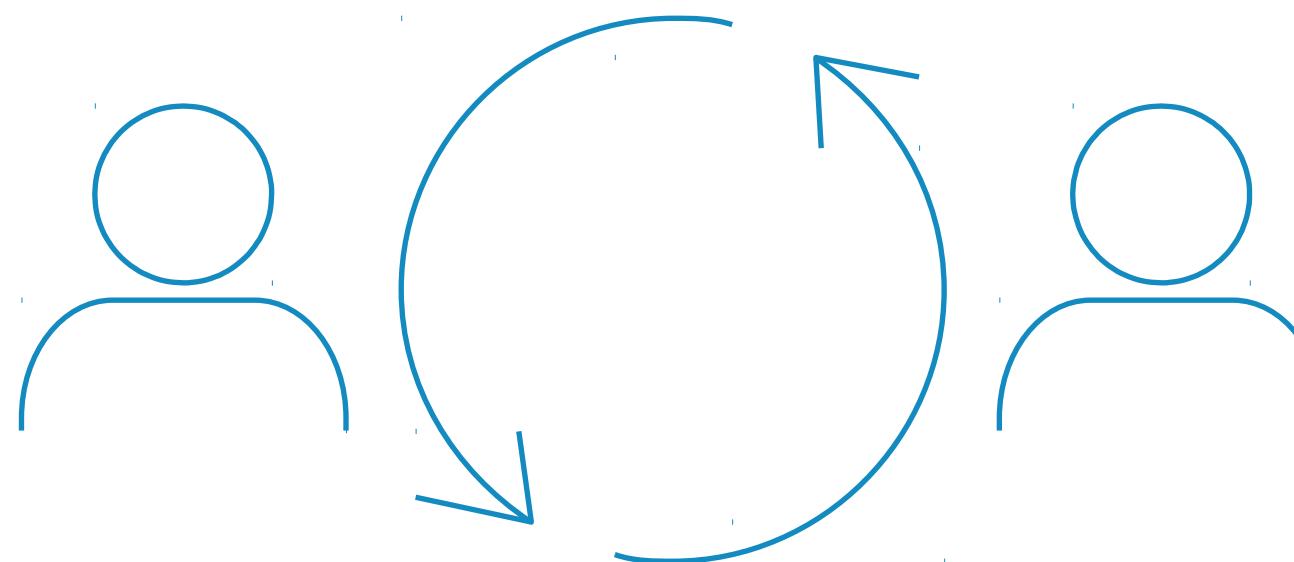
Business



A BETTER WAY TO SHARE DATA



INSTANT, LIVE DATA SHARING



- No data movement; nothing is copied
- Data consumers immediately see all updates
- Share with an unlimited number of consumers



DATA SHARING ACCOUNTS



Data Providers

Share data with others



Data Consumers

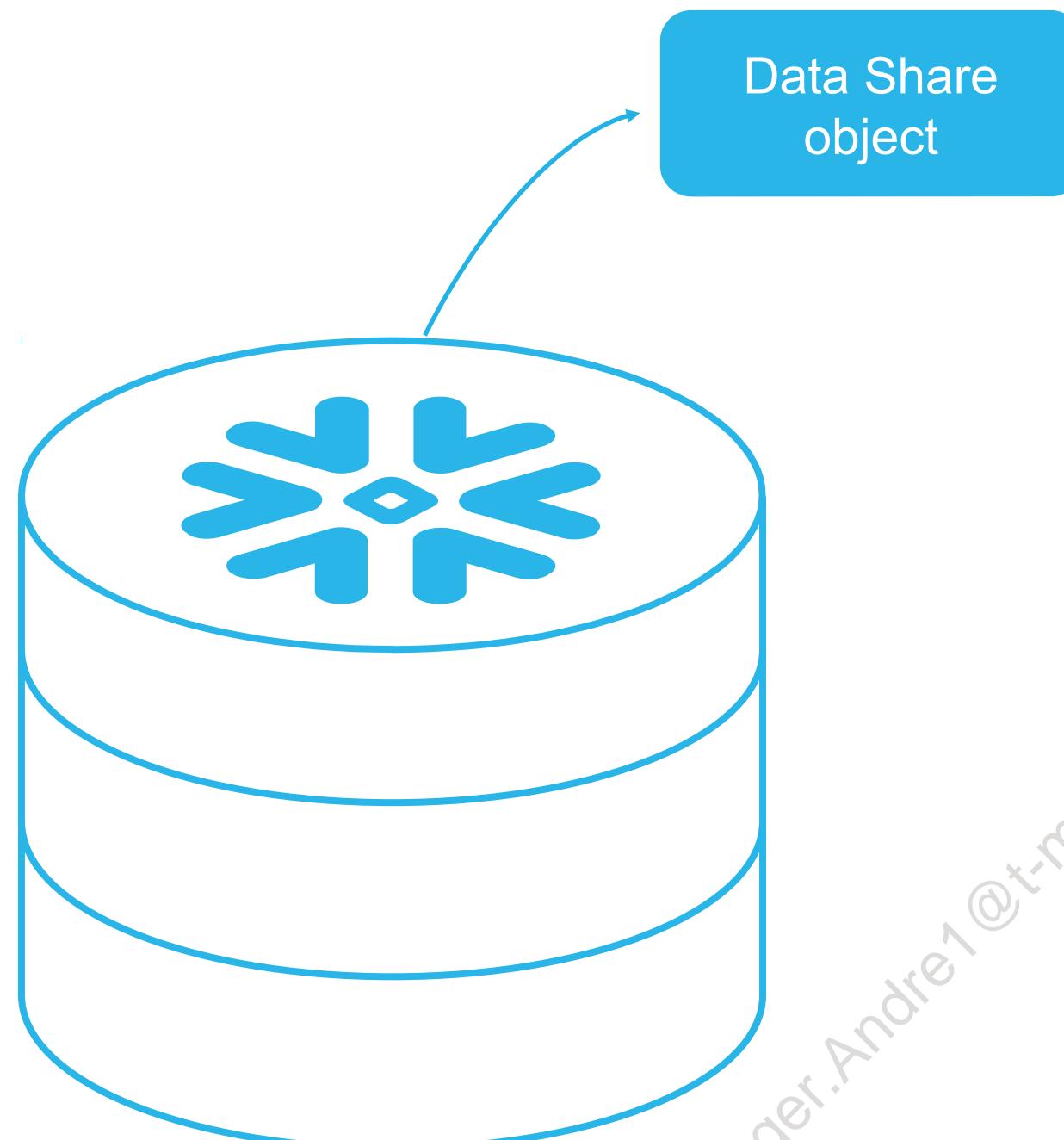
Accesses shared data with
their own Snowflake
account



Reader Accounts

Query data using
compute from data
provider's account

WHAT IS A SHARE?



- A named Snowflake object that encapsulates all the information required to share objects within a database
- Share Contains:
 - Privileges that grant access to the database, schema, or specific objects
 - Data Consumer or Reader account name



ABOUT SHARES

- Shares are read-only
- Tables, secure views, and secure UDFs can be shared
- Access to a share can be revoked at any time
- Consumers can create new tables from a share
- Shares are limited to a single region and cloud provider. Database replication can be used to share to another region or cloud provider.
- When sharing to an account with a lower version of Snowflake (EX: Business Critical to Enterprise), an override will be required to allow the share.



SHARING SECURE VIEWS

WITH ACCESS CONTROLS



Data
Provider



Data
Consumers

id	partner_data		acct_map	
	sales	commission	id	acct_name
100	2,350	11.75	100	ABC
100	1,975	49.375	200	XYZ
200	3,459	8.6475		
200	9,156	68.67		

Account = ABC

Account = XYZ



SHARING SECURE VIEWS

WITH ACCESS CONTROLS



Data
Provider



Data
Consumers

id	partner_data		acct_map	
	sales	commission	id	acct_name
100	2,350	11.75	100	ABC
100	1,975	49.375	200	XYZ
200	3,459	8.6475		
200	9,156	68.67		

Account = ABC

Account = XYZ

```
CREATE SECURE VIEW shared_data
AS SELECT * FROM partner_data pd
JOIN acct_map am ON pd.id = am.id
AND am.acct_name = CURRENT_ACCOUNT();
```



SHARING SECURE VIEWS

WITH ACCESS CONTROLS



Data Provider



Data Consumers

id	partner_data		acct_map	
	sales	commission	id	acct_name
100	2,350	11.75	100	ABC
100	1,975	49.375	200	XYZ
200	3,459	8.6475		
200	9,156	68.67		

```
CREATE SECURE VIEW shared_data
AS SELECT * FROM partner_data pd
JOIN acct_map am ON pd.id = am.id
AND am.acct_name = CURRENT_ACCOUNT();
```

```
GRANT SELECT ON shared_data TO SHARE shr_sales;
```



Account = ABC

Account = XYZ

SHARING SECURE VIEWS

WITH ACCESS CONTROLS



Data Provider



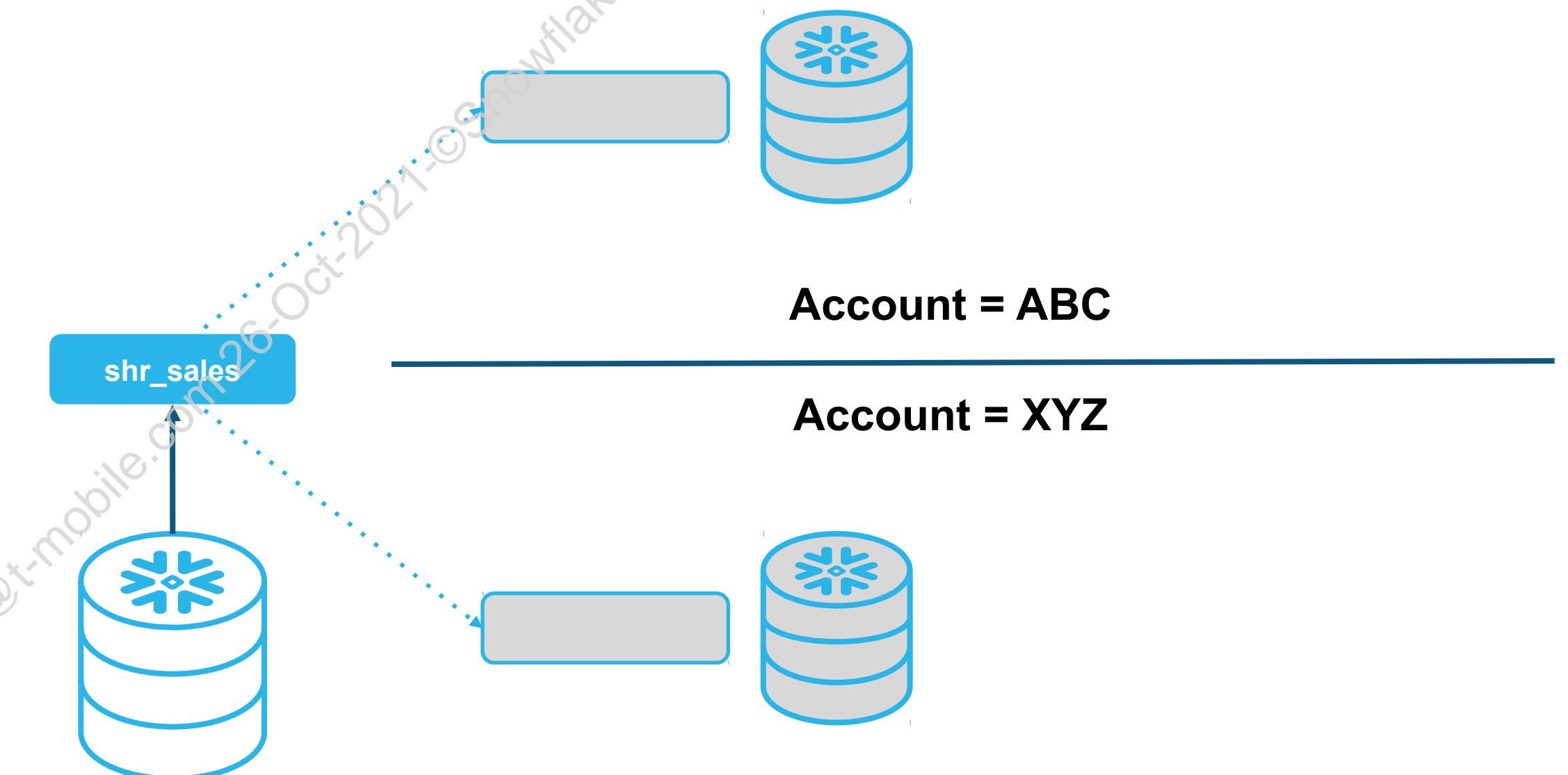
Data Consumers

id	partner_data		acct_map	
	sales	commission	id	acct_name
100	2,350	11.75	100	ABC
100	1,975	49.375	200	XYZ
200	3,459	8.6475		
200	9,156	68.67		

```
CREATE SECURE VIEW shared_data
AS SELECT * FROM partner_data pd
JOIN acct_map am ON pd.id = am.id
AND am.acct_name = CURRENT_ACCOUNT();
```

```
GRANT SELECT ON shared_data TO SHARE shr_sales;
```

```
ALTER SHARE shr_sales ADD ACCOUNTS=ABC,XYZ;
```



SHARING SECURE VIEWS

WITH ACCESS CONTROLS



Data Provider



Data Consumers

id	partner_data		acct_map	
	sales	commission	id	acct_name
100	2,350	11.75	100	ABC
100	1,975	49.375	200	XYZ
200	3,459	8.6475		
200	9,156	68.67		

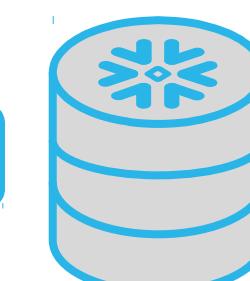
```

CREATE SECURE VIEW shared_data
AS SELECT * FROM partner_data pd
JOIN acct_map am ON pd.id = am.id
AND am.acct_name = CURRENT_ACCOUNT();

GRANT SELECT ON shared_data TO SHARE shr_sales;

ALTER SHARE shr_sales ADD ACCOUNTS=ABC,XYZ;

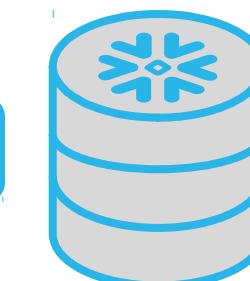
```



id	shared_data	
	sales	commission
100	2,350	11.75
100	1,975	49.375

Account = ABC

Account = XYZ



id	shared_data	
	sales	commission
200	3,459	8.6475
200	9,156	68.67



DATA PROVIDERS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA SHARING ACCOUNTS



Data Providers

Share data with others



Data Consumers



Reader Accounts



DATA PROVIDERS



Data Providers



© 2021 Snowflake Computing Inc. All Rights Reserved

- Snowflake account that creates shares and makes them available for others to consume
- Unlimited number of shares can be created, and an unlimited number of accounts can be added to a share
- Grants provide granular access control to selected objects, including at the row level (using filters)



Data Provider
(ProvXyz)

```
USE ROLE ACCOUNTADMIN;

CREATE SHARE share1;                                --empty share

GRANT USAGE ON DATABASE sales TO SHARE share1;      -- add database
GRANT USAGE ON SCHEMA sales.east TO SHARE share1;   -- add schema
GRANT SELECT ON TABLE east.accts TO SHARE share1;   -- add table

ALTER SHARE share1 ADD ACCOUNTS=Cons123;           -- add account
```



CONSIDERATIONS FOR PROVIDERS

- Create shares with a role that has CREATE SHARES privileges (such as the ACCOUNTADMIN role)
- Consumer accounts must be in the same region and cloud provider.
- Consumer accounts at a lower version than the provider will require an override.
- Use database replication to easily cross regions and/or cloud platforms.
- Only tables, secure views, and secure UDFs are supported in shares at this time.
- New or modified data in a share are immediately available to all consumers.
- You must grant usage on new objects created in a database in a share in order for them to be available to consumers.



Data Providers



DATA CONSUMERS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



DATA SHARING ACCOUNTS



Data Providers



Data Consumers

Accesses shared data with
their own Snowflake
account



Reader Accounts

DATA CONSUMERS



Data Consumers

- Snowflake account that accesses a share from another account
- Create a local database to "hold" the share
 - Can only be done once
- Can consume an unlimited number of shares
- Are charged for their own compute on that share





Data Consumer
(Consz123)

CONSUME A SHARE

```
USE ROLE ACCOUNTADMIN;
```

```
CREATE DATABASE From_provider
FROM SHARE ProvXyz.share1;          --read-only shared database
```

```
USE DATABASE From_provider;        --switch to read-only database
```

```
SELECT * FROM east.accts;          --same as querying any other
                                         database in your account
```



CONSIDERATIONS FOR CONSUMERS

- Administer shares with a role that has IMPORT SHARES privileges (such as the ACCOUNTADMIN role)
- Share can only be consumed once per account
- Shared databases are read-only
- Shared data can be copied into a table in the consumer account (but cannot be cloned)
- You cannot forward (re-share) shared databases
- Time travel is not supported for shared databases



Data Consumers

INSTRUCTOR DEMO

Data Sharing

15 minutes



THE DATA MARKETPLACE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

INSTRUCTOR DEMO

Data Marketplace

15 minutes



READER ACCOUNTS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

DATA SHARING ACCOUNTS



Data Providers



Data Consumers



Reader Accounts

Query data using
compute from data
provider's account

Roger.Andre1@tmobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

READER ACCOUNTS



Reader Accounts

- Consumer who does not already have a Snowflake account
- Accounts are created by providers
- Storage and compute credits funded by provider
- Consumer cannot write data to the account unless permission is granted by the provider
- Allows consumer to experiment with data sharing without a Snowflake contract



CONSIDERATIONS FOR READER ACCOUNTS

- Requires additional configuration after creating the share and the reader account
 - Create a database from the share
 - Configure Users, Roles, Security for access to data in the reader account
 - Create Virtual Warehouses for Reader use (paid for by the Data Provider)
 - Set default role and warehouses for logins
 - Set up a resource monitor
- Provider responsible for support for the reader accounts



Data Reader



SET UP A READER ACCOUNT

Data Provider
(ProvXyz)

```
USE ROLE ACCOUNTADMIN;
```

```
CREATE SHARE share1;                                --empty share
```

```
GRANT USAGE ON DATABASE sales TO SHARE share1;      -- add database
```

```
GRANT USAGE ON SCHEMA sales.east TO SHARE share1;   -- add schema
```

```
GRANT SELECT ON TABLE east.accts TO SHARE share1;   -- add table
```

```
CREATE MANAGED ACCOUNT <name>                  -- create reader account
```

```
ADMIN_NAME=<username>, ADMIN_PASSWORD=<pw>,  
TYPE=READER;
```

```
ALTER SHARE share1 ADD ACCOUNTS=<account>;        -- add account
```

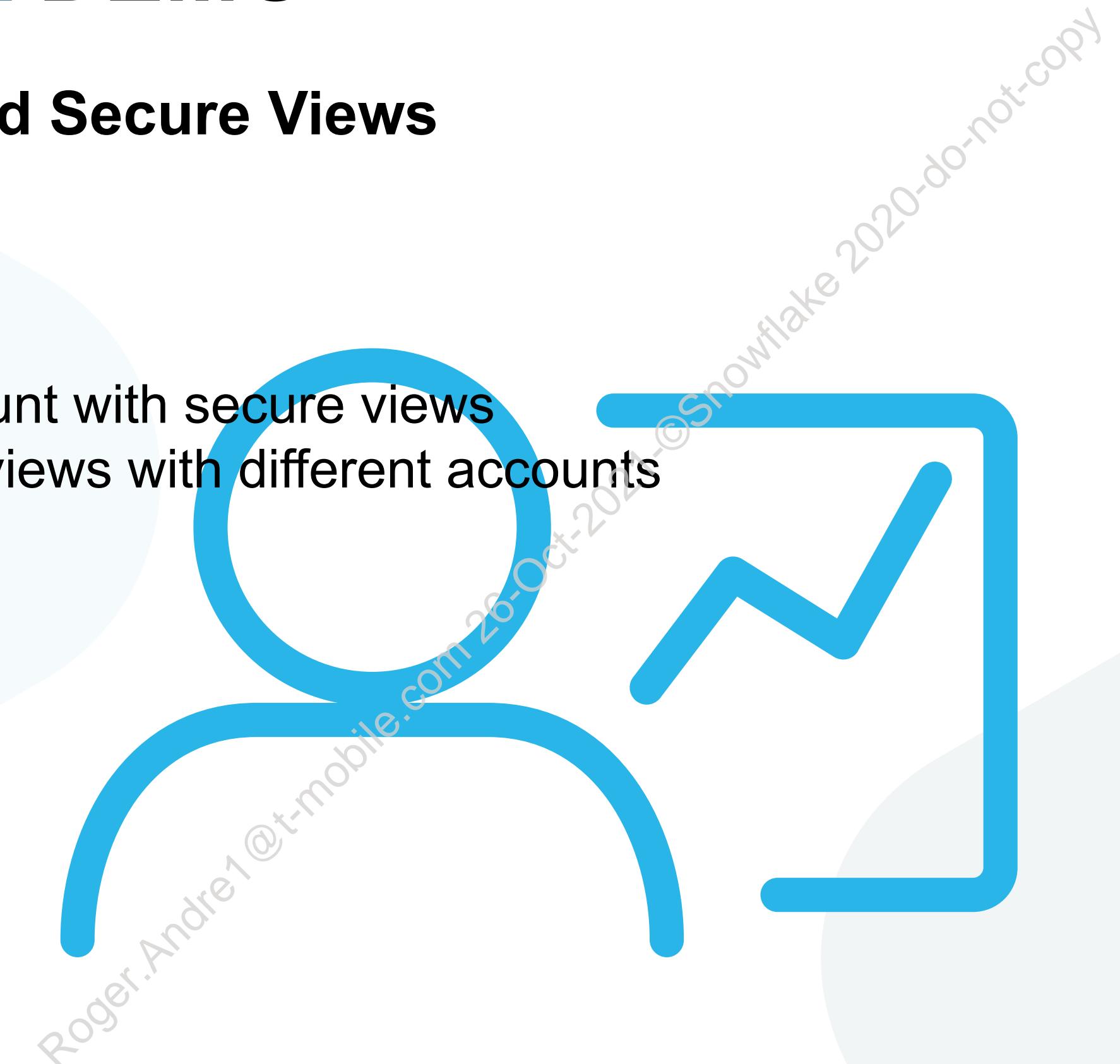


INSTRUCTOR DEMO

Reader Accounts and Secure Views

20 minutes

- Create a share
- Create a reader account with secure views
- Demonstrate secure views with different accounts



PERFORMANCE & CONCURRENCY

Roger.Andre1@t-mobile.com
26-Oct-2021
©Snowflake 2020-do-not-copy



MODULE AGENDA

- SQL Performance Tips
- Resolving Pruning Issues
- Virtual Warehouse Scaling

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



SQL PERFORMANCE TIPS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



TYPICAL DATABASE ORDER OF EXECUTION



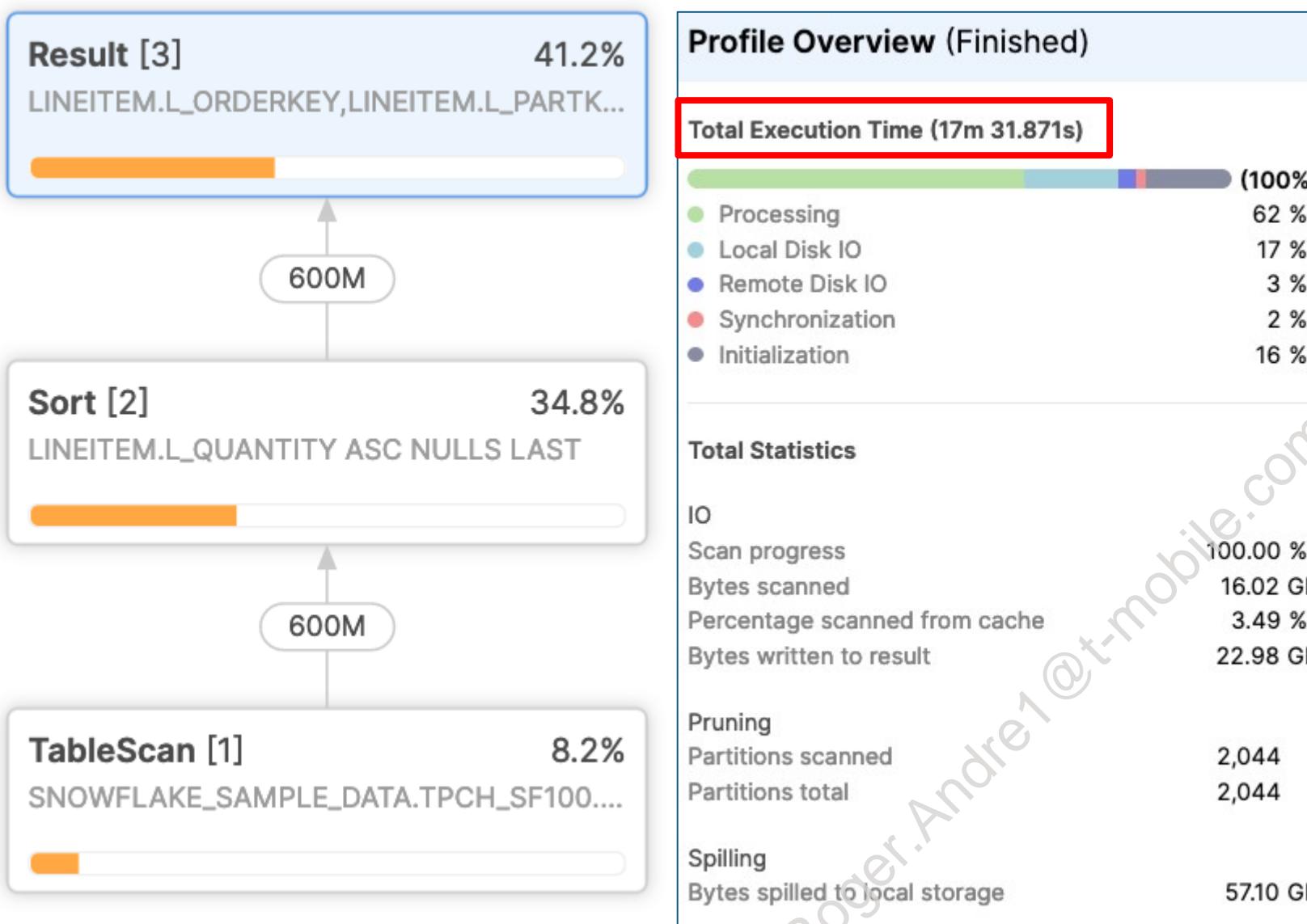
TOP PERFORMANCE TIP

- Row operations are performed before GROUP operations
- Check row operations first
 - First, check FROM and WHERE clauses
 - Then, check GROUP BY and HAVING clauses
- Use appropriate filters, as early as possible
 - Filters do help where applicable
 - The Optimizer uses the filters to prune out unnecessary data

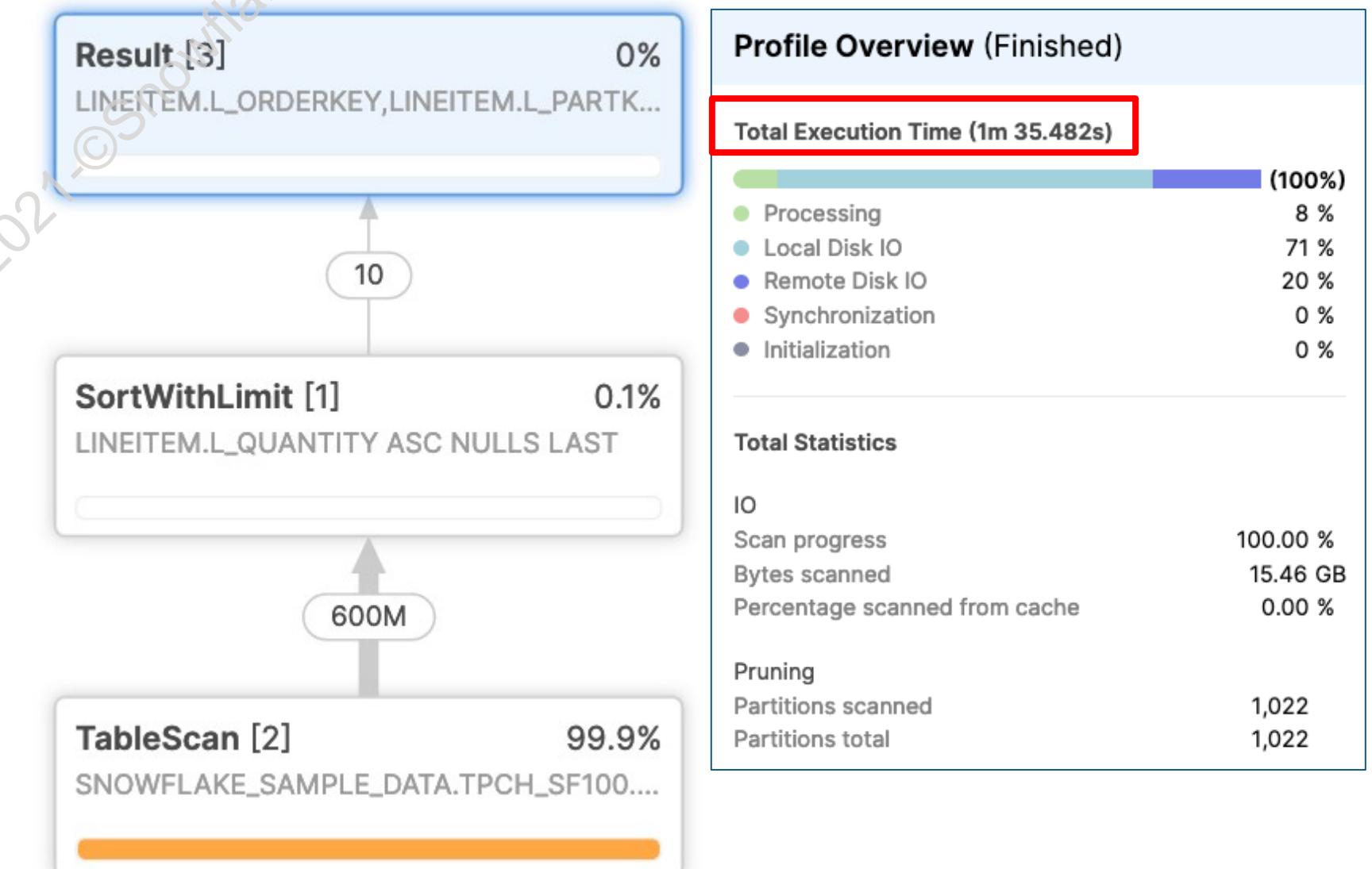


ADD LIMIT TO LARGE ORDER BY

```
SELECT * FROM LINEITEM  
ORDER BY L_QUANTITY;
```

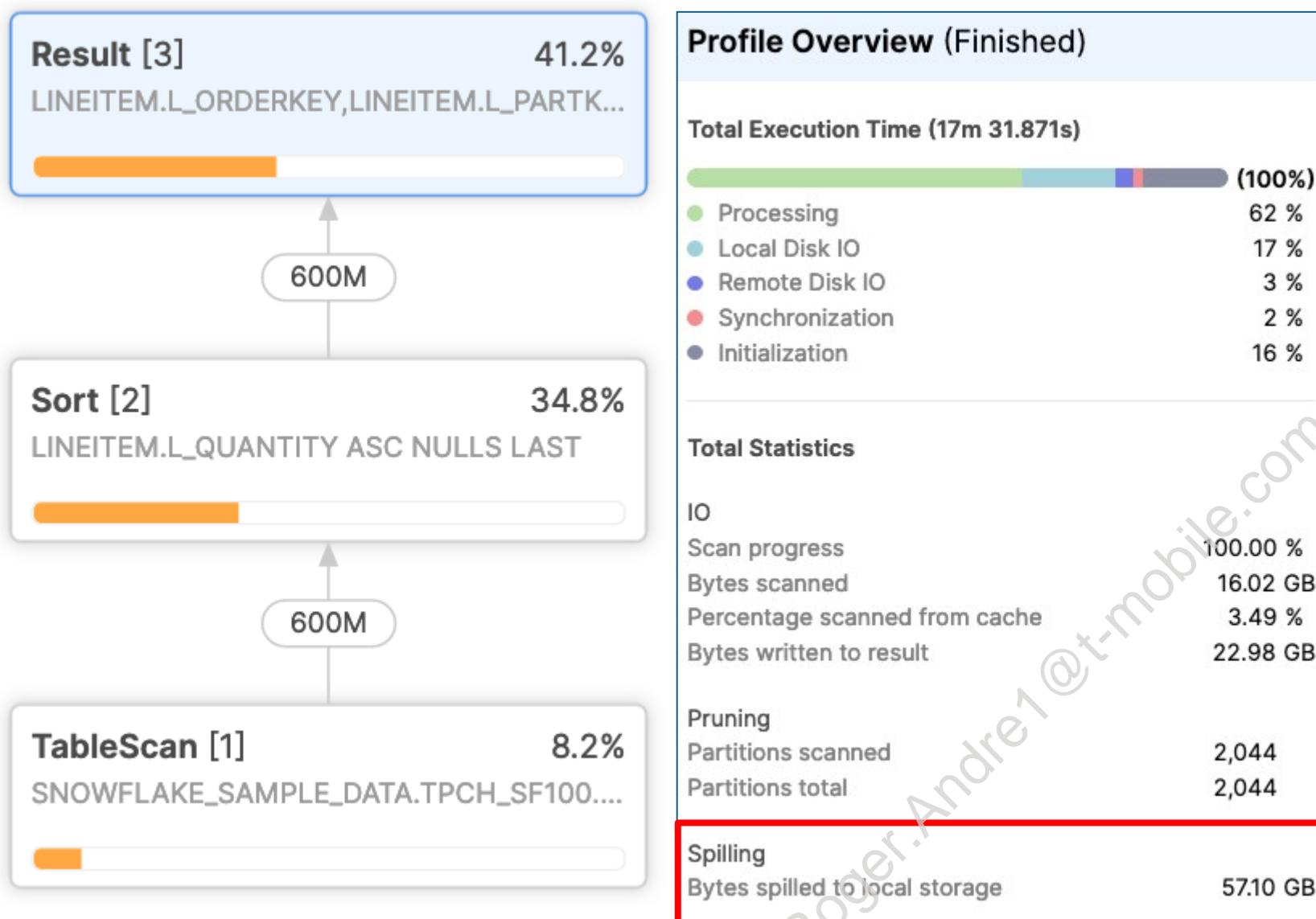


```
SELECT * FROM LINEITEM  
ORDER BY L_QUANTITY LIMIT 10;
```

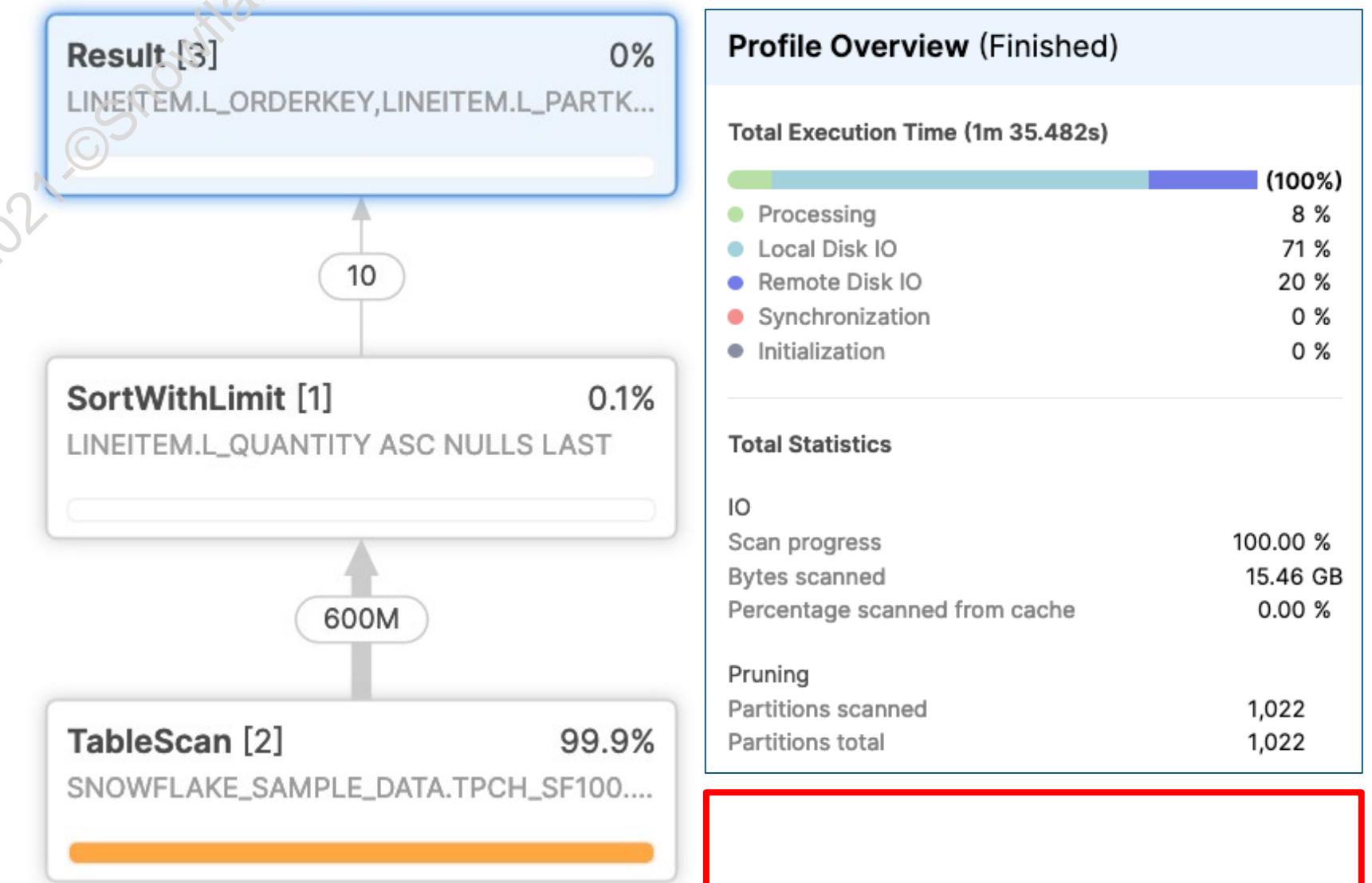


ADD LIMIT TO LARGE ORDER BY

```
SELECT * FROM LINEITEM  
ORDER BY L_QUANTITY;
```

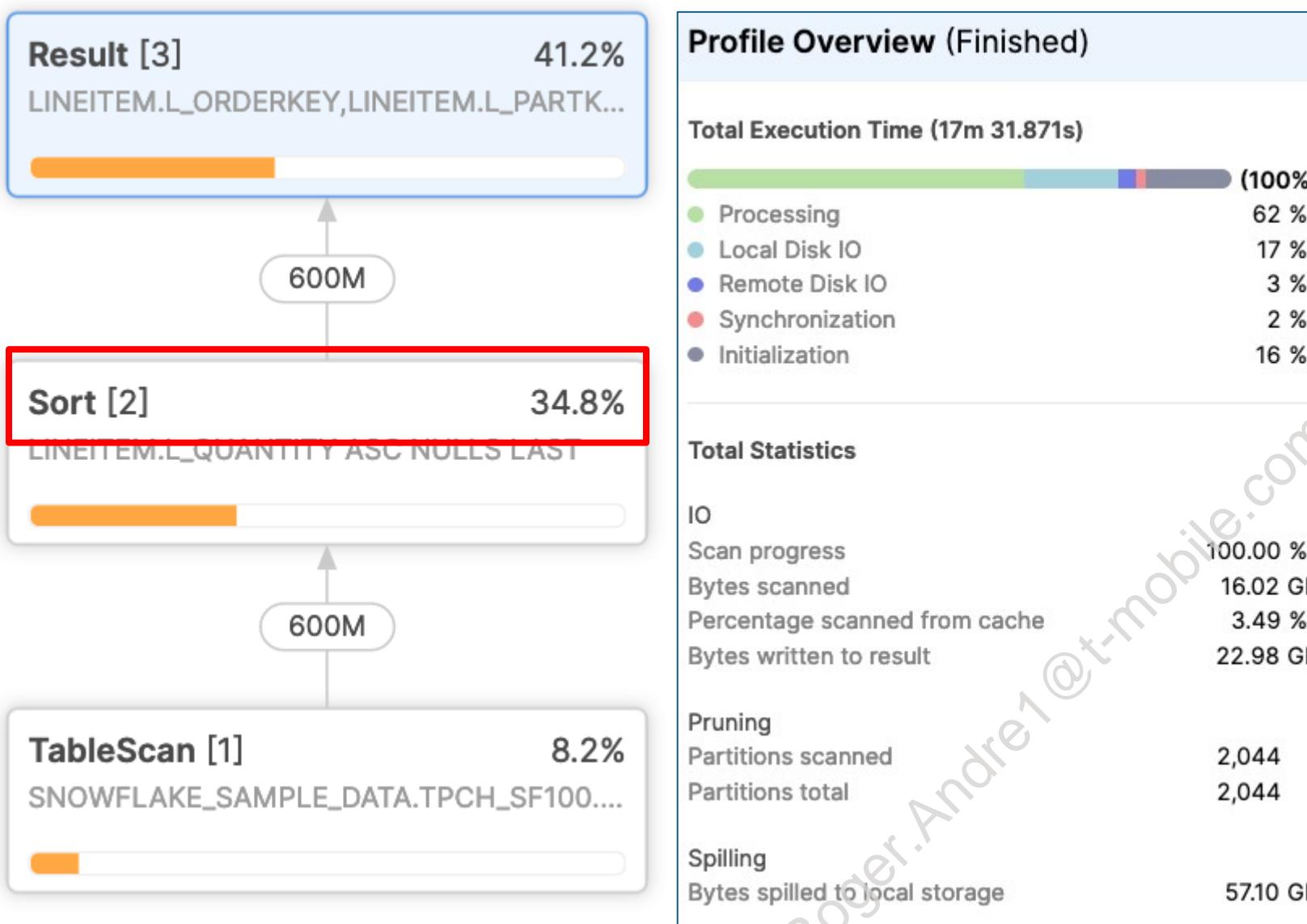


```
SELECT * FROM LINEITEM  
ORDER BY L_QUANTITY LIMIT 10;
```

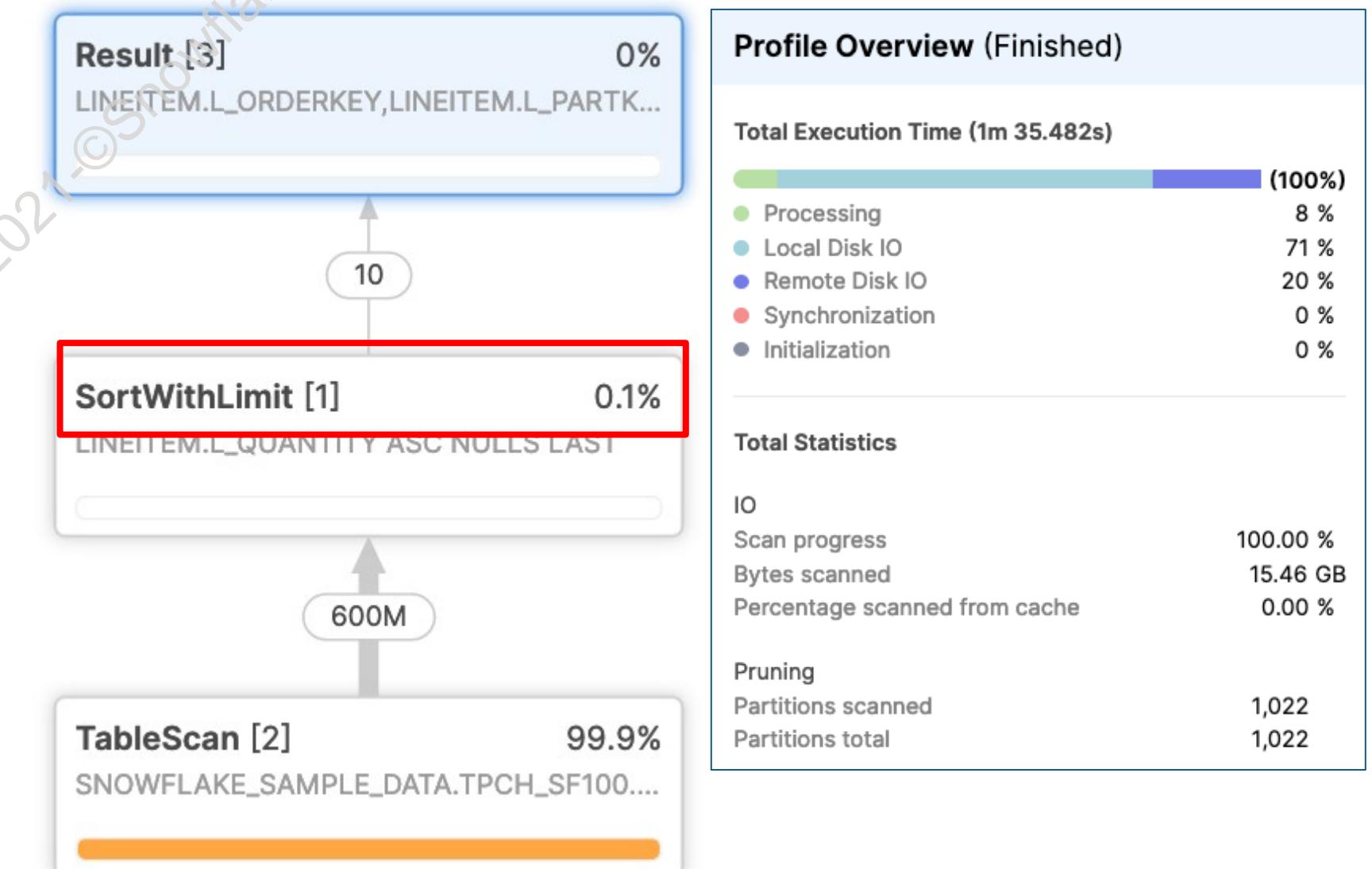


ADD LIMIT TO LARGE ORDER BY

```
SELECT * FROM LINEITEM  
ORDER BY L_QUANTITY;
```



```
SELECT * FROM LINEITEM  
ORDER BY L_QUANTITY LIMIT 10;
```



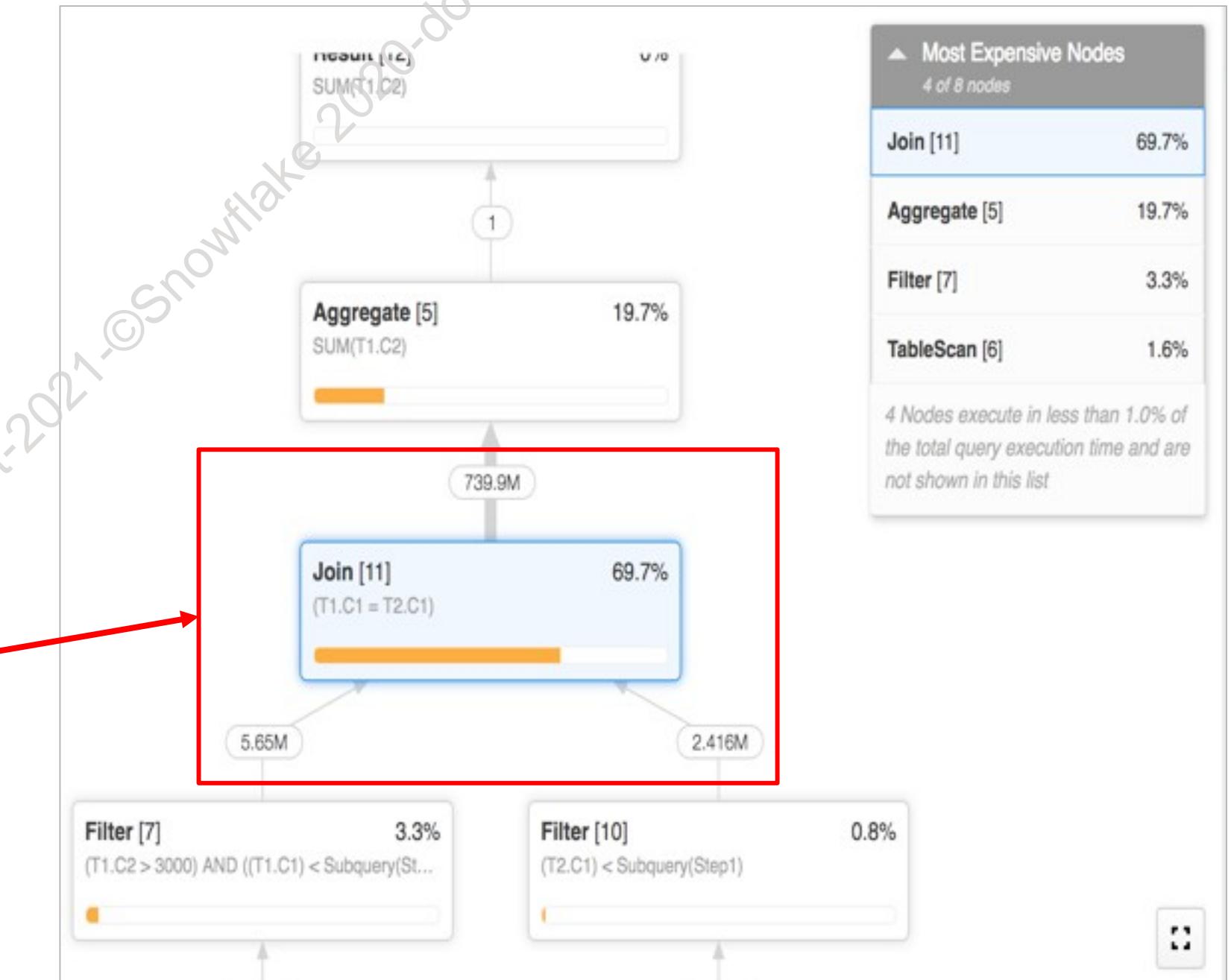
JOIN ON UNIQUE KEYS

Best Practices

- Ensure keys are distinct
- Understand relationships between your tables before joining
- Avoid many-to-many join
- Avoid unintentional cross join

Troubleshooting Scenario

- Joining on non-unique keys can explode your data output (**join explosion**)
 - Each row in table1 matches multiple rows in table 2



SNOWFLAKE BUILT-IN OPTIMIZATIONS

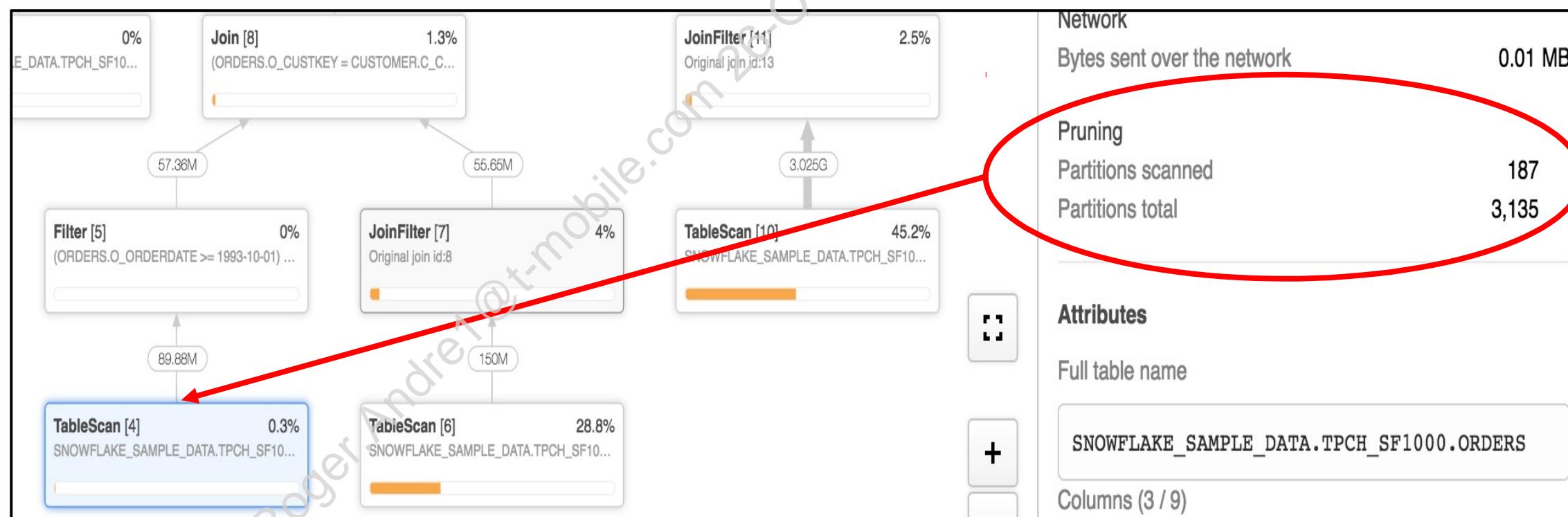
- Snowflake provides patented micro-partition pruning to optimize query performance:
 - Static partition pruning based on columns in WHERE clause
 - Dynamic partition pruning based on JOIN columns of a query
- Best practices to assist SQL optimizer:
 - Apply appropriate filters as early as possible in the query
 - For naturally clustered tables, apply appropriate predicate columns (e.g., date columns) that have a high correlation to ingestion order
- Smart Scan
 - Covered in the Performance Workshop



PARTITION PRUNING

Effective pruning: query's filter column matches table's clustering order

```
SELECT <items>
FROM order
WHERE o_orderdate >= to_date('1993-10-01')
AND o_orderdate < dateadd(month, 3, to_date('1993-10-01'));
```



PREDICATES AND PERFORMANCE

- Built-in functions and UDFs are very useful, but they can impact performance when used in a query predicate

```
SELECT l_orderkey
FROM lineitem l, orders o
WHERE l_orderkey=o_orderkey AND
      LOG(10, l_extendprice) > 4.5 AND
      LOG(10, o_totalprice - l_tax) > 4.5
```

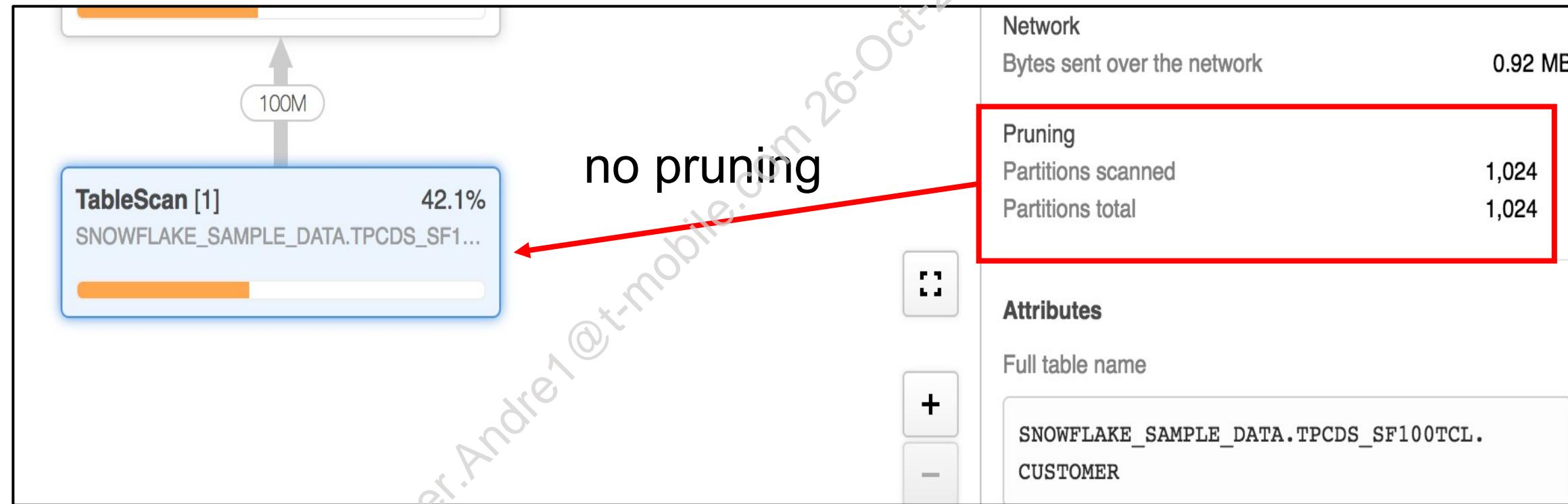
- Consider materializing the intermediate result using a temporary table



PITFALLS OF NON-PERFORMING PREDICATES

Some WHERE predicates provide no opportunity to optimize pruning

```
SELECT c_customer_id, c_last_name  
FROM tpcds.customer  
WHERE UPPER (c_last_name) LIKE '%KROLL%'
```



UNCORRELATED VS CORRELATED SUBQUERIES

- Uncorrelated scalar subqueries (with no external column references)

```
SELECT p.name, p.annual_wage, p.ctry FROM pay AS p
WHERE p.annual_wage <
  (SELECT per_capita_gdp FROM intl_gdp
WHERE country = 'Brazil');
```

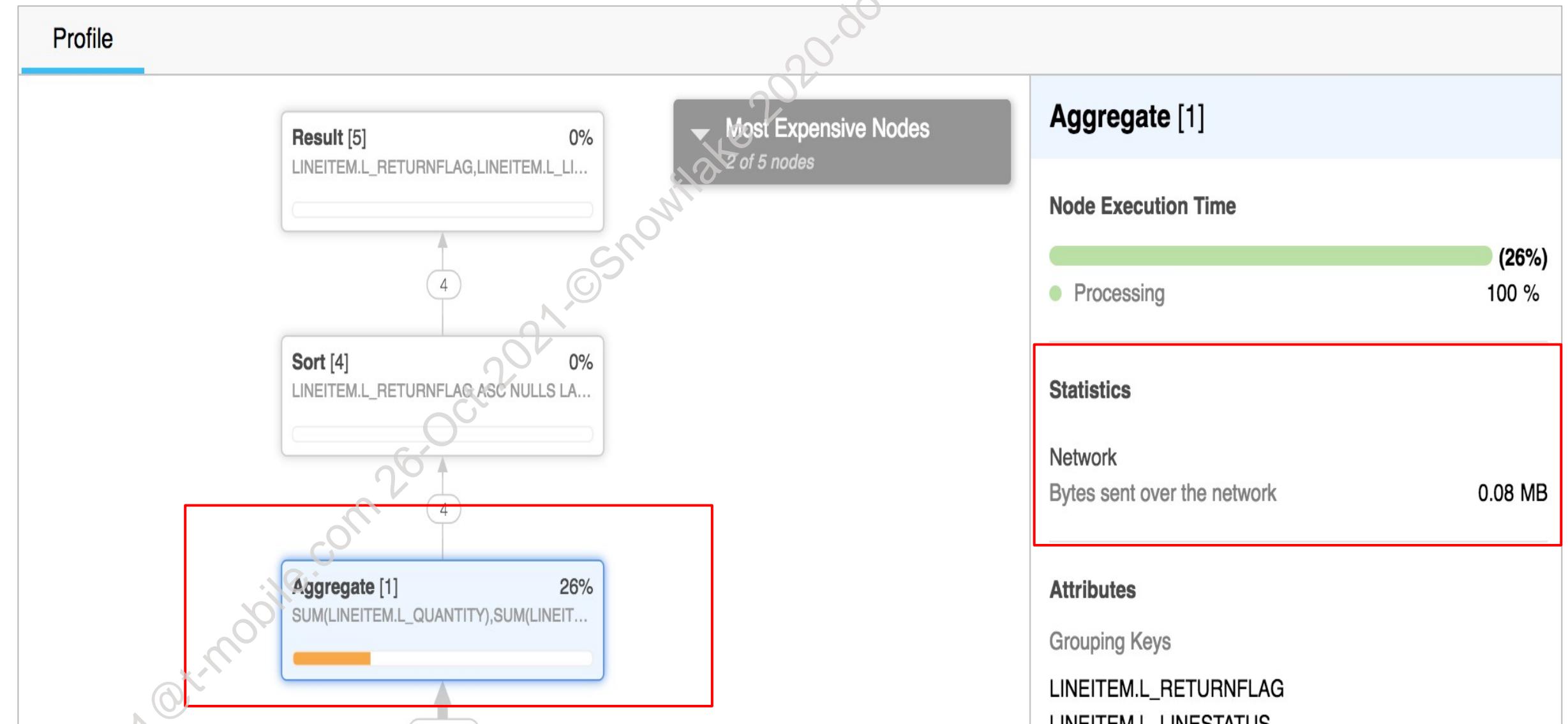
- Correlated scalar subqueries in WHERE clause

```
SELECT p.name, p.annual_wage, p.ctry FROM pay AS p
WHERE p.annual_wage <
  (SELECT MAX(per_capita_gdp) FROM intl_gdp i
WHERE p.ctry = i.country);
```



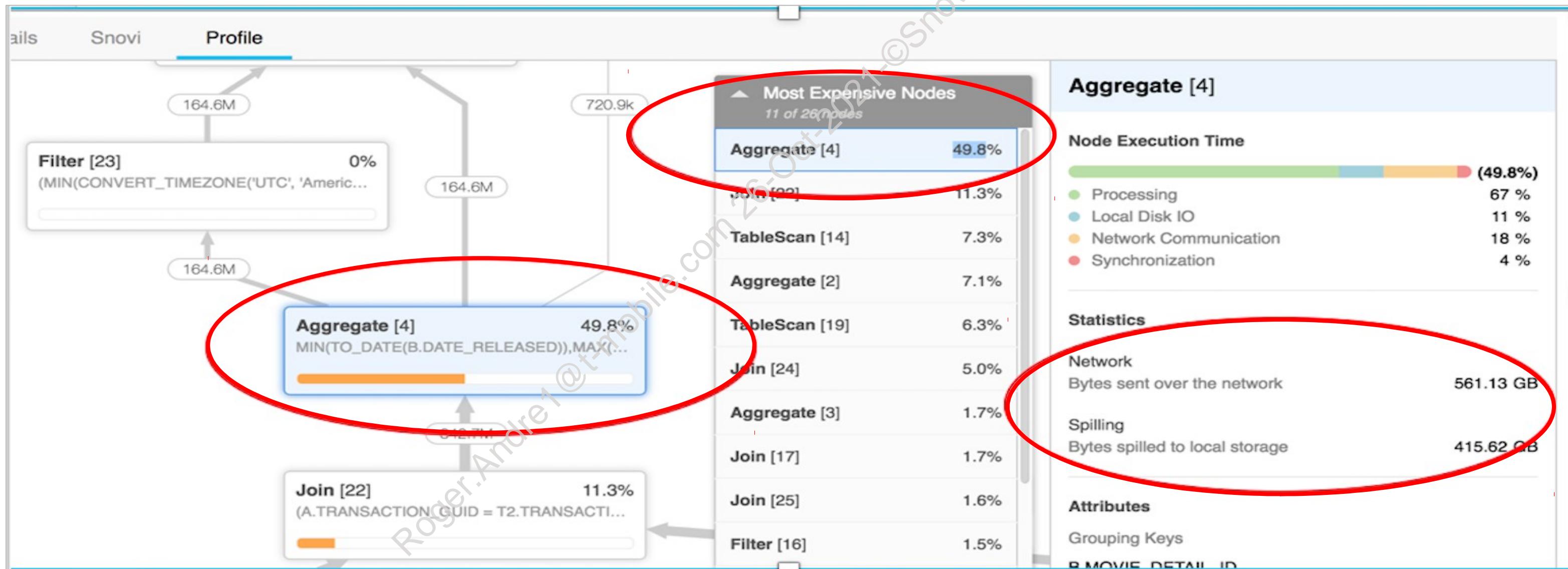
GROUP BY WITH FEW DISTINCT VALUES

```
SELECT  
    l_returnflag,  
    l_linenstatus,  
    SUM(l_quantity)  
FROM lineitem  
GROUP BY  
l_returnflag,  
l_linenstatus;
```



GROUP BY WITH MANY DISTINCT VALUES

- Memory-intensive
- Spilling to disk, high network data
- Sub-optimal performance



ORDER BY

```
select
    o_orderpriority,
    count(*) as order_count
from
    orders
where
    o_orderdate >= to_date('1993-07-01')
    and o_orderdate < dateadd(month, 3, to_date('1993-07-01'))
    and exists (
        select
            *
        from
            lineitem
        where
            l_orderkey = o_orderkey
            and l_commitdate < l_receiptdate
            order by l_orderkey
    )
group by
    o_orderpriority
order by
    o_orderpriority;
```

Wasted Compute

Best Practice

- Orders values in ascending or descending order on a specified column
- Use ORDER BY in top level SELECT only
 - ORDER BY in subqueries does not impact the result, and slows performance



RESOLVING PRUNING ISSUES

Roger.Andre1@t-mobile.com 26-Oct-2021 © Snowflake 2020-do-not-copy



NATURAL DATA CLUSTERING

- As data is loaded into tables, micro-partitions are created based on ingestion time
 - This data is “naturally clustered” i.e., organized by order of ingestion
- Ingestion date may highly correlate with one or more columns
 - Tables with a sequential field
 - Tables with a date field

ID	NICKNAME	YOB
1	PRISCILLA	1962
2	DAWG	2001
3	JOKER	1997
4	PRINCESS	1998
5	SNEEZY	1983

DATE	ORDER_NUMBER	ITEM_TYPE
22-Jul-2019 23:29:07	2019_072111356	Office Supplies
22-Jul-2019 23:32:00	2019_072111357	Office Furniture
22-Jul-2019 23:44:56	2019_072111358	Services
22-Jul-2019 23:59:01	2019_072111359	Office Supplies
23-Jul-2019 00:01:07	2019_072111360	Printers



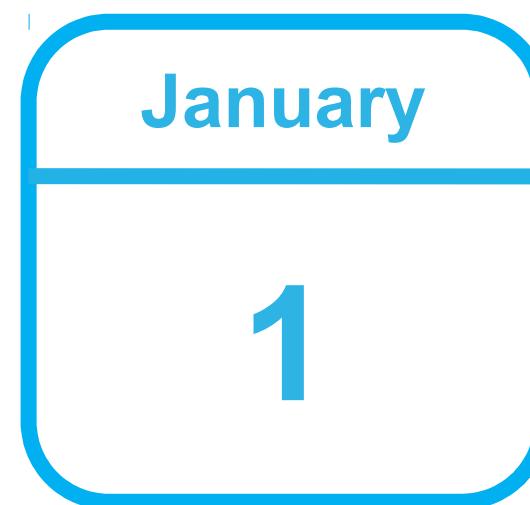
WHAT DETERMINES NATURAL CLUSTERING?

- A single data load reads source data and writes it into some number of micro-partitions
- The source data organization determines what range of values are represented in the micro-partitions
- Examples:
 - Source data contains rows from a specific day: that day's data will be in contiguous micro-partitions
 - Source data contains rows for a month or year: those micro-partitions will have high/low values for dates within that month or year
 - Source data is ordered alphabetically on a “last name” column: contiguous micro-partitions will be in alphabetical order



PRUNING ISSUES / NATURAL DATA CLUSTERING

- Resolving pruning issues requires an understanding of clustering
- As data is loaded into tables, micro-partitions are created based on ingestion order
- Micro-partitions are co-located based on ingestion order
- All tables have an inherent or “natural” clustering



Ingested data ordered
by day

Year	Month
2020	01
2020	02
2020	03
2021	01
2021	02

Ingested data ordered
by year, month

Garza, Kate
Johnson, Bob
Okada, Tim
Smith, Larry
Washington, LaTonya

Ingested data ordered
by last name

COMMON EXAMPLES

Tables with a sequential field

ID	NICKNAME	YOB
1	PRISCILLA	1962
2	DAWG	2001
3	JOKER	1997
4	PRINCESS	1998
5	SNEEZY	1983

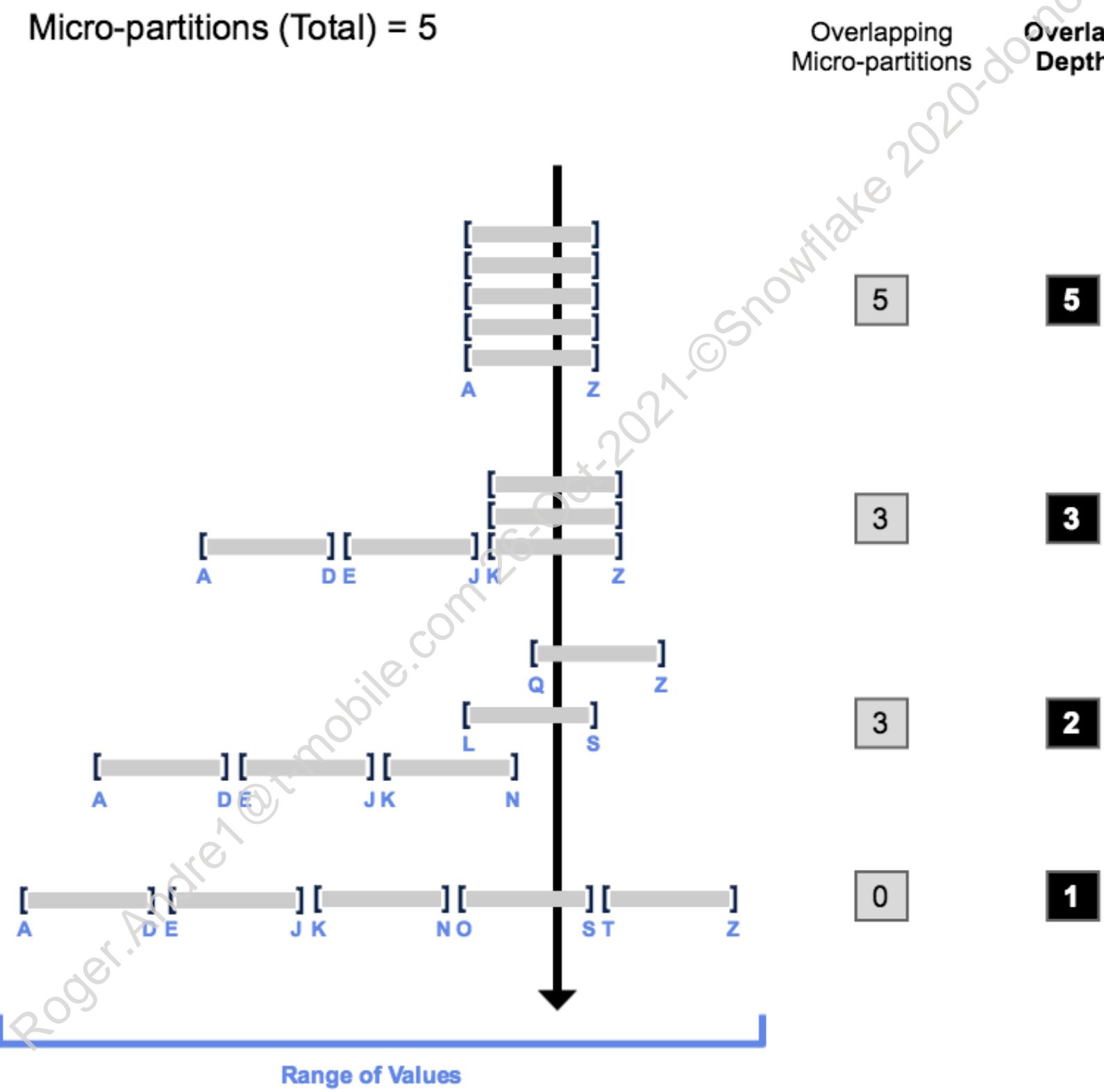
Tables with a date field

DATE	ORDER_NUMBER	ITEM_TYPE
22-Jul-2019 23:29:07	2019_072111356	Office Supplies
22-Jul-2019 23:32:00	2019_072111357	Office Furniture
22-Jul-2019 23:44:56	2019_072111358	Services
22-Jul-2019 23:59:01	2019_072111359	Office Supplies
23-Jul-2019 00:01:07	2019_072111360	Printers

- Natural clustering becomes less efficient as rows are inserted, deleted, or updated
- Snowflake's micro-partitioning and metadata is usually sufficient to keep queries performant



CLUSTERING DEPTH ILLUSTRATED



OPTIONS

- Divide tables into sub-tables (i.e., a table for a month, year, or quarter)
- Create a materialized view with a subset of the table
- Change the natural clustering order by changing the ingestion order
- Apply a clustering key



CLUSTERING KEYS

- If needed, specific columns can be made “clustering keys” to improve query performance
- Candidate tables for clustering keys:
 - Tables in the multi-terabyte range that change infrequently
 - Tables whose query performance has degraded noticeably over time
- After clustering, like data (by key) is co-loaded in the same micro-partitions
- Snowflake keeps the clustering order updated automatically, billed on a per-second basis
- Use 1-3 keys maximum, in order of low to high cardinality
 - More than 3 keys usually costs more than the benefit gained



WHAT TABLES SHOULD BE CLUSTERED?

- Clustering keys are not for every table!
 - Automatic clustering consumes credits
 - Automatic reclustering also increases storage costs
 - Clustering and reclustering can be made less expensive if a cluster key is added after loading the table
 - Automatic clustering can be disabled
- Good candidates for clustering keys:
 - Tables in the multi-terabyte range experience the most benefit from clustering (particularly if DML is performed regularly on these tables)
 - Tables that change infrequently are less expensive to re-cluster
 - Tables whose query performance degrades noticeably over time



VIRTUAL WAREHOUSE SCALING

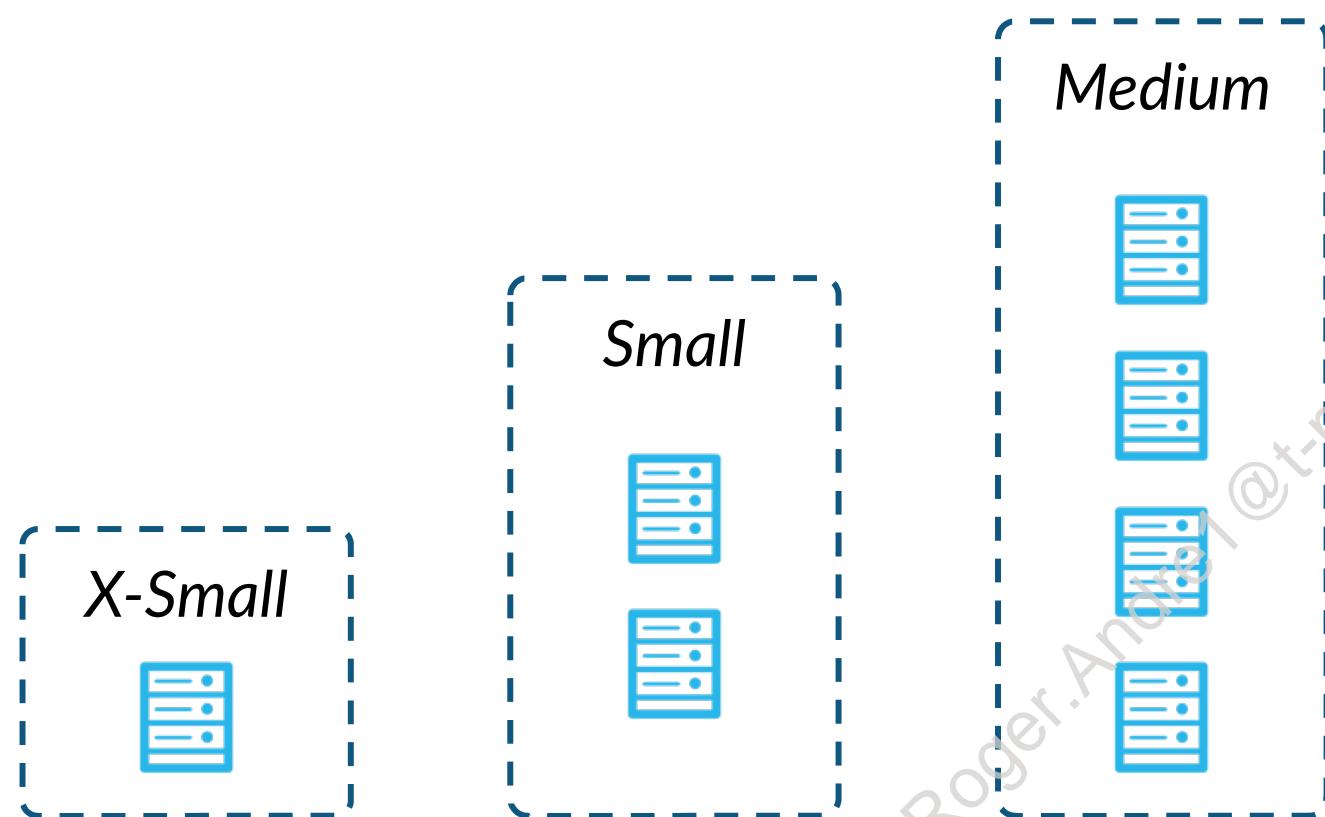
Roger.Andre1@t-mobile.com 26-Oct-2021 ©snowflake 2020-do-not-copy



SCALING UP VS. SCALING OUT

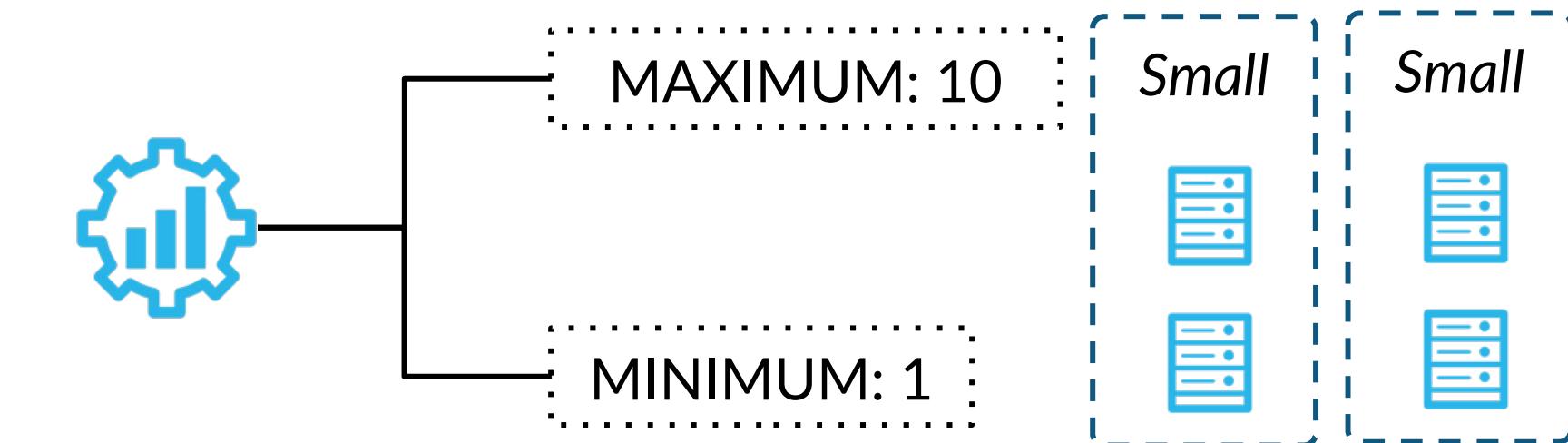
SCALING UP

Resizing a warehouse to handle complex/process-intensive queries



SCALING OUT

Adding more clusters to your current Virtual Warehouse *without* changing the size of the Warehouse to handle concurrency issues.



SERVERS PER CLUSTER

Warehouse Size	Servers	Clusters
X-Small	1	1
Small	2	1
Medium	4	1
Large	8	1
X-Large	16	1
2X-Large	32	1
3X-Large	64	1
4X-Large	128	1

- As queries are submitted, required resources are calculated and reserved
- If there are insufficient resources, that query is queued until other queries finish and release resources



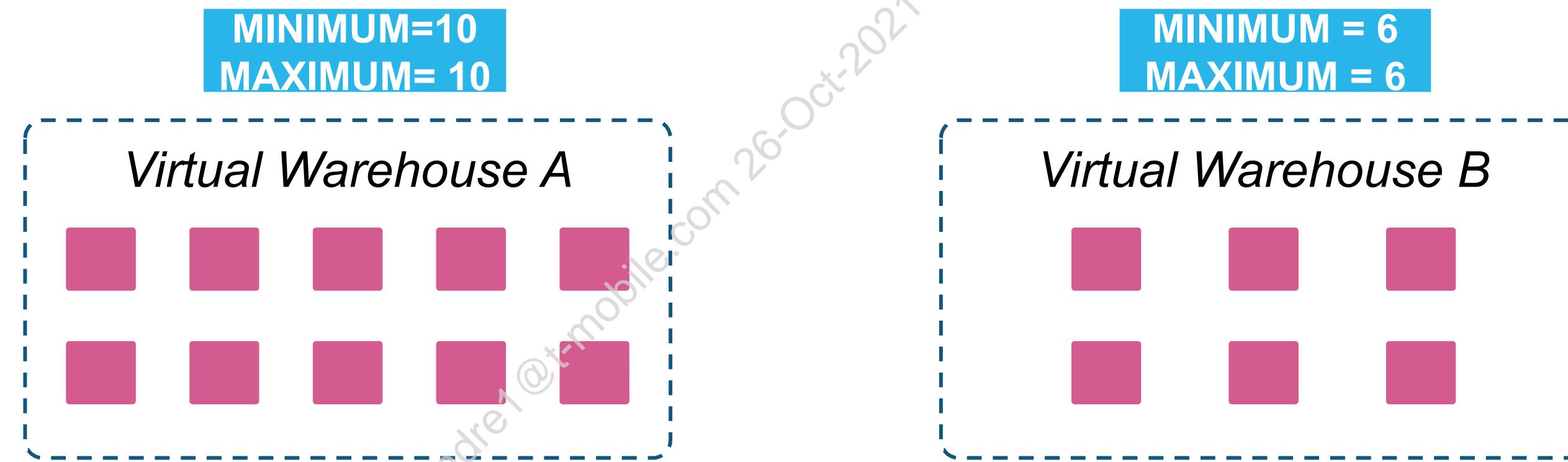
SCALING OUT: MULTI-CLUSTER WAREHOUSES

- Multi-cluster warehouses are designed specifically for handling queuing and performance issues related to large numbers of concurrent users and/or queries.
- Multi-cluster warehouses can help automate this process if your number of users/queries tend to fluctuate.



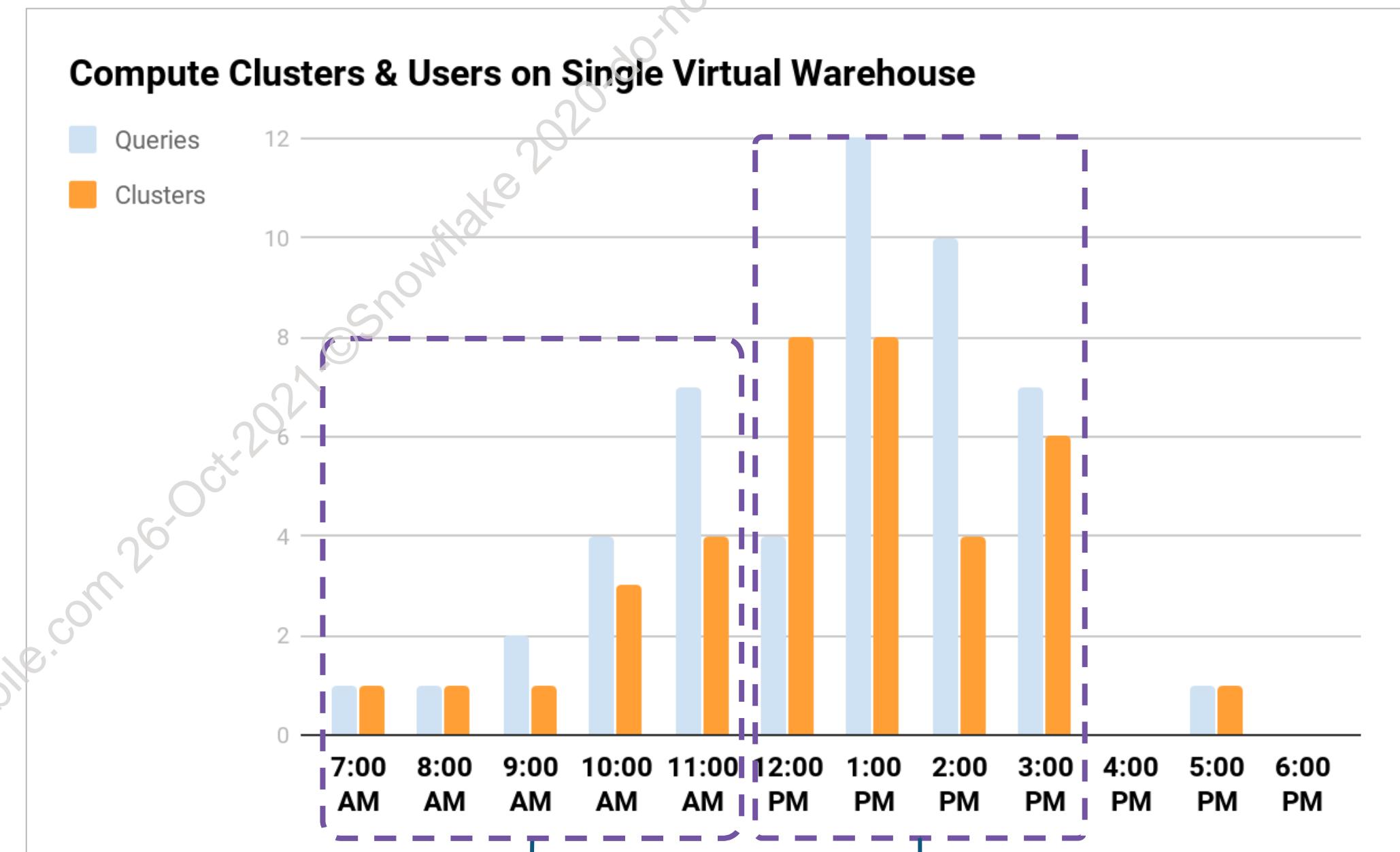
SCALING OUT: MAXIMIZED

- When the warehouse is started, Snowflake starts all the clusters so that maximum resources are available while the warehouse is running



SCALING OUT: AUTO-SCALING

Allows Snowflake to start and stop clusters as needed to dynamically manage the load on the warehouse

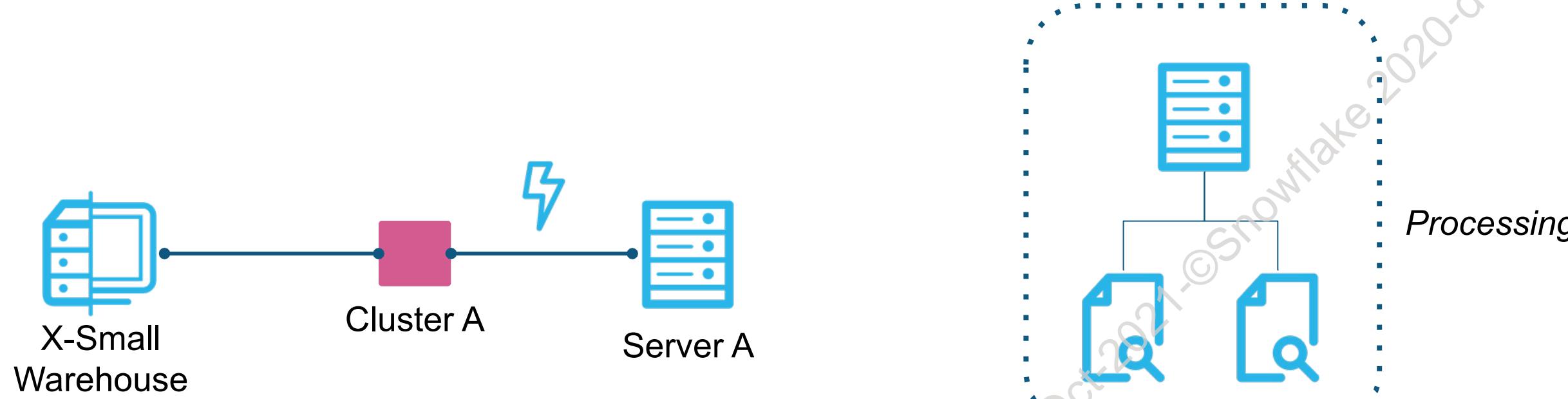


AUTO-SCALING: SCALING POLICY

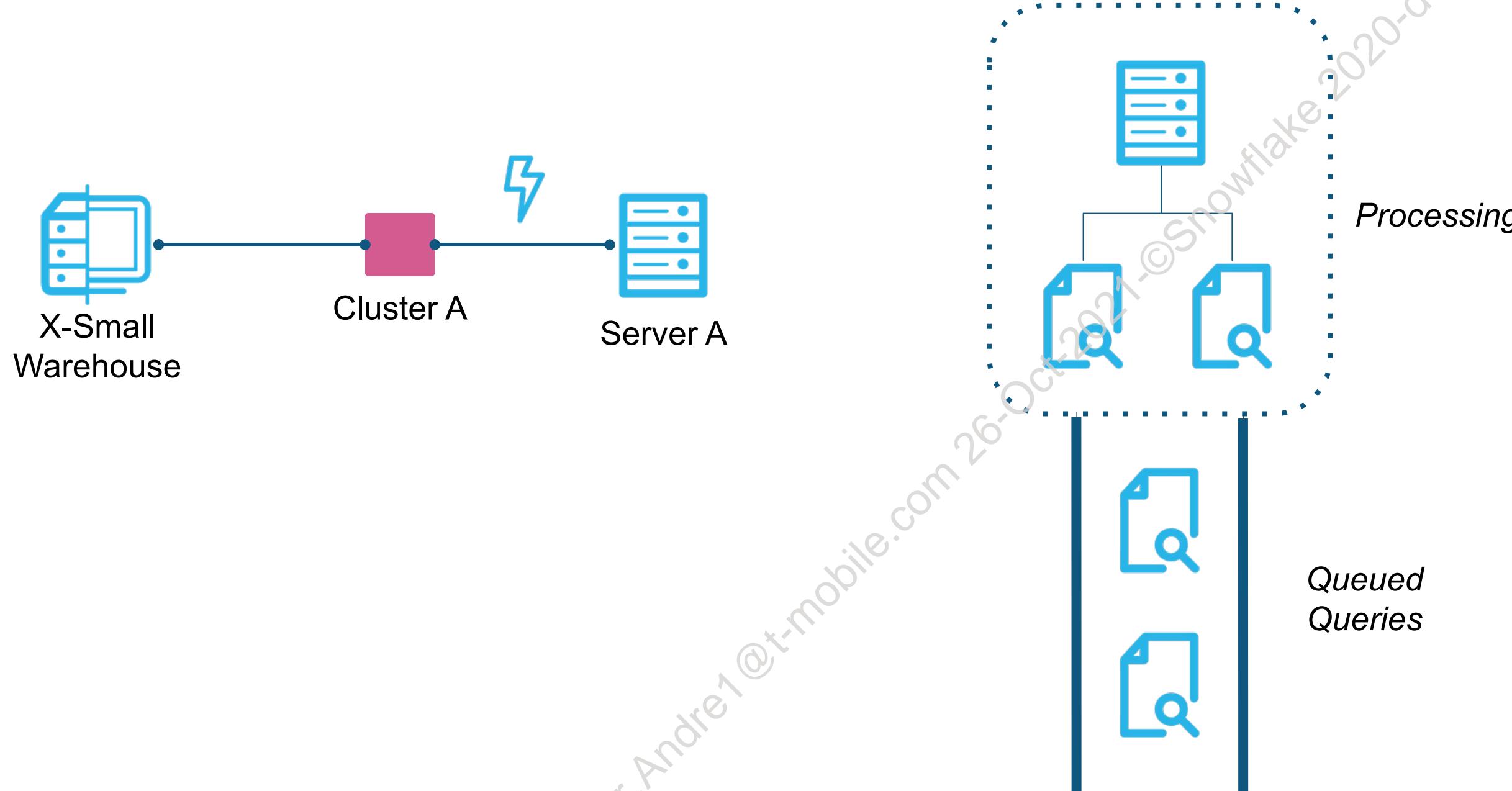
Policy	Description	Cluster Starts...	Cluster Shuts down...
Standard	Minimizes queuing and starts additional clusters.	Immediately when either a query is queued or the system detects that there's one more query than the currently-running clusters can execute.	After 2-3 consecutive checks (performed at 1 minute intervals) determine that the load on the least-loaded cluster could be redistributed to the others without spinning up the cluster again.
Economy	Favors keeping running clusters fully-loaded rather than starting additional clusters; could result in jobs being queued rather than starting additional Clusters.	Only if the system estimates there's enough query load to keep the cluster busy for at least 6 minutes.	After 5-6 consecutive checks (performed at 1 minute intervals), determine that the load on the least-loaded cluster could be redistributed to the others without spinning up the cluster again.



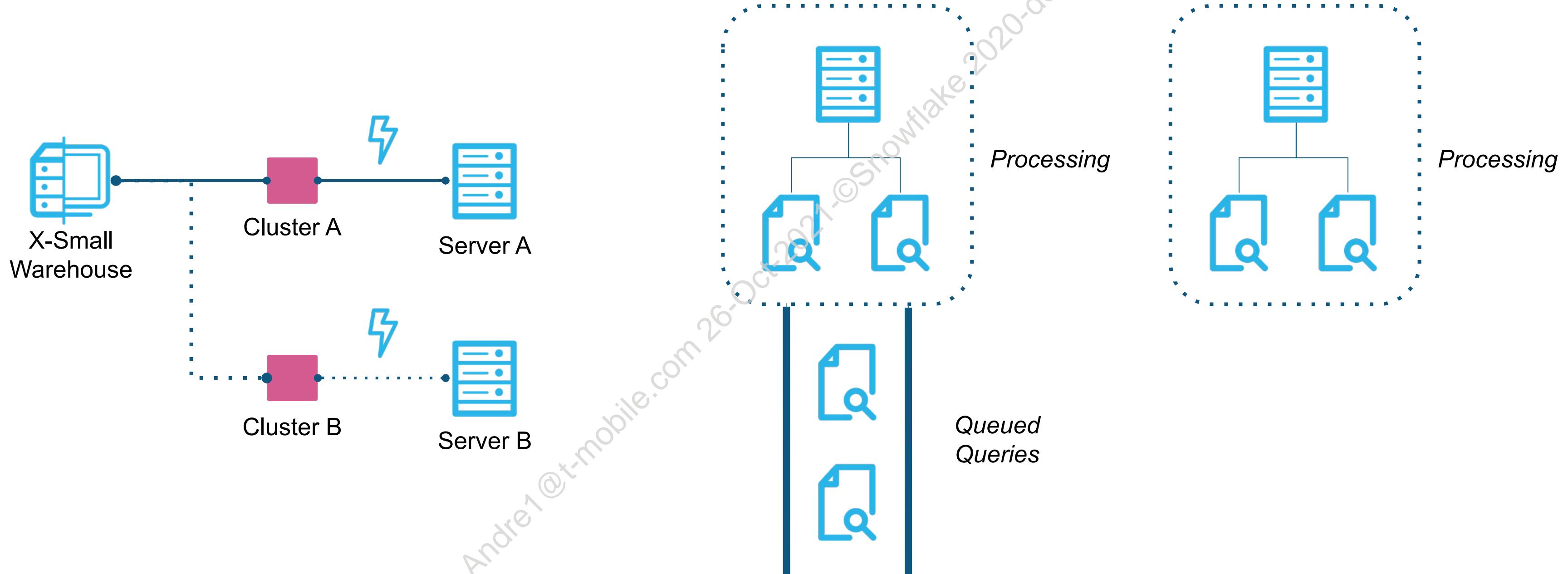
SCALING OUT POLICY: STANDARD



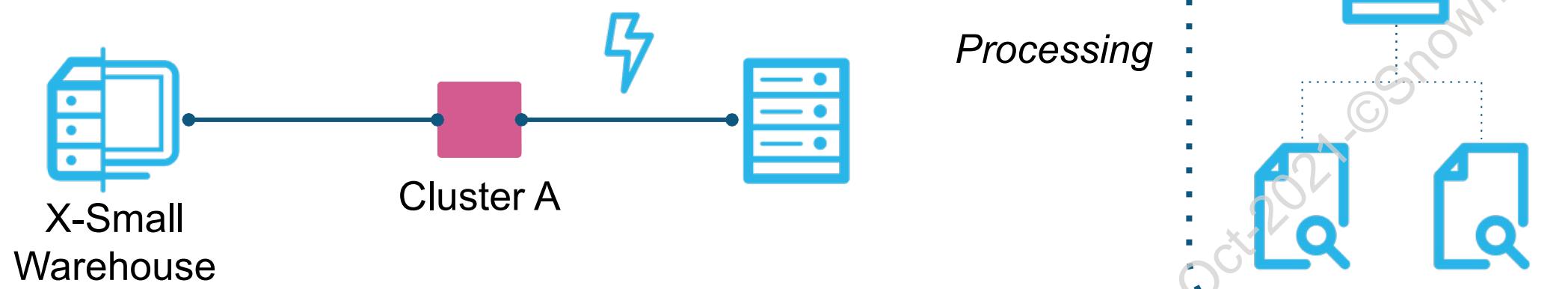
SCALING OUT POLICY: STANDARD



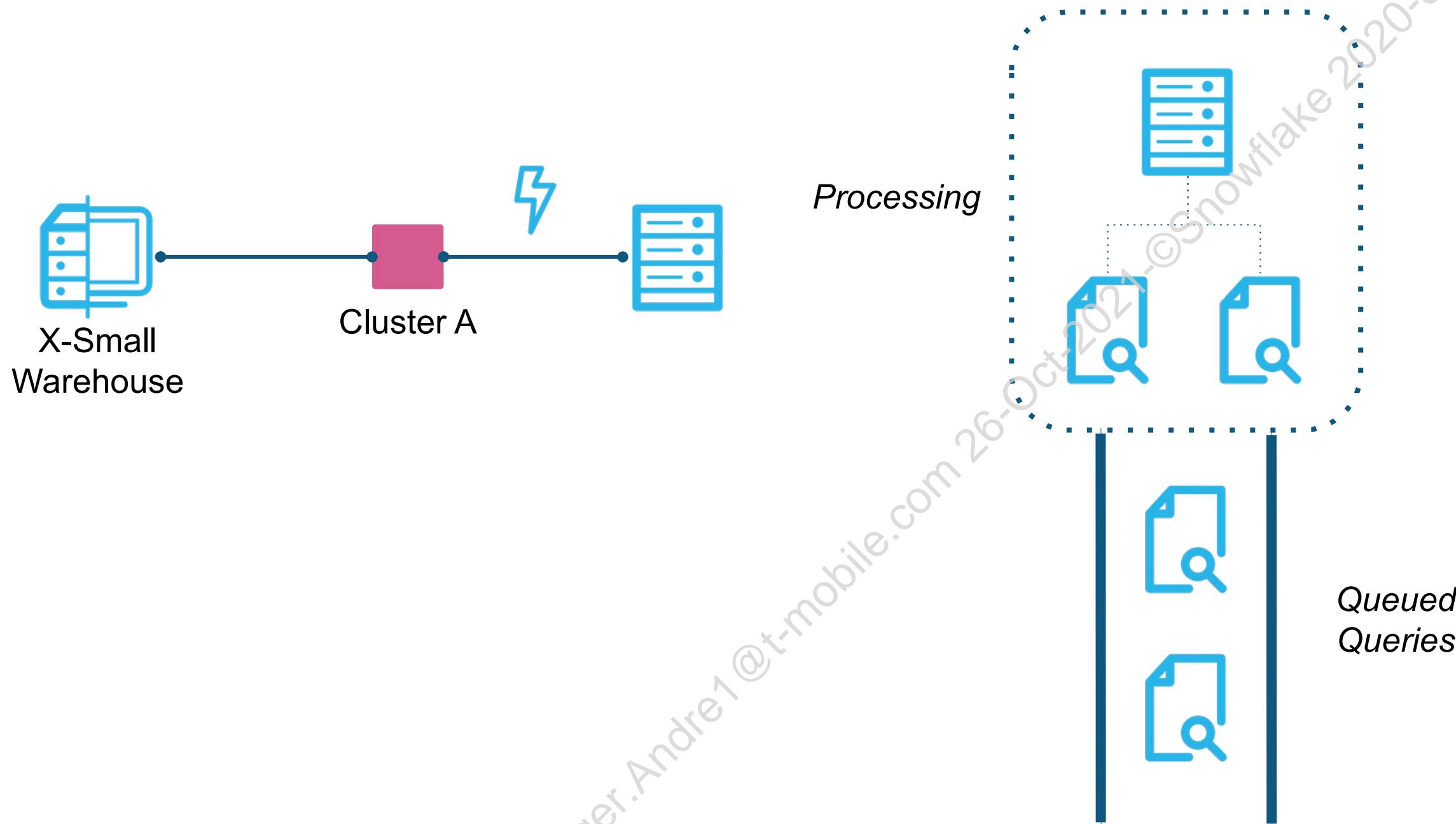
SCALING OUT POLICY: STANDARD



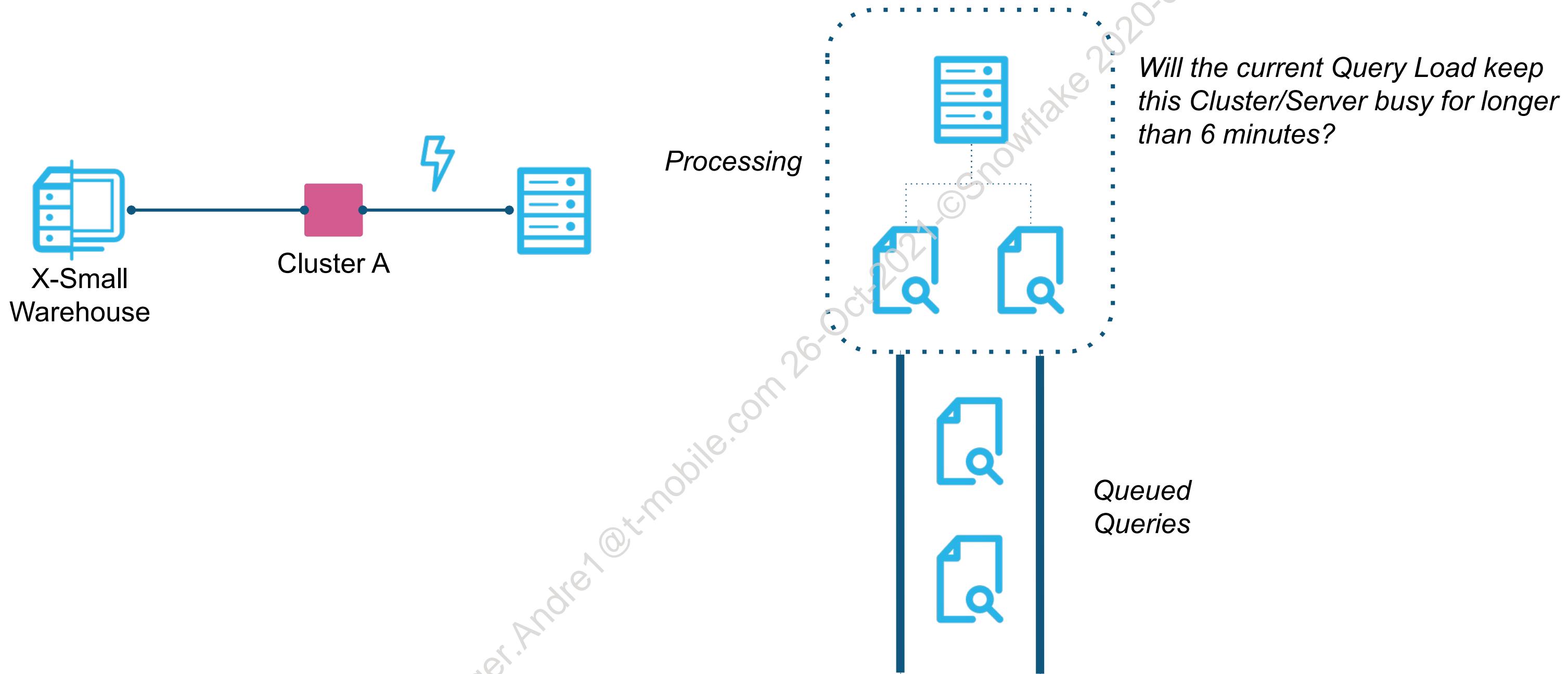
SCALING OUT POLICY: ECONOMY



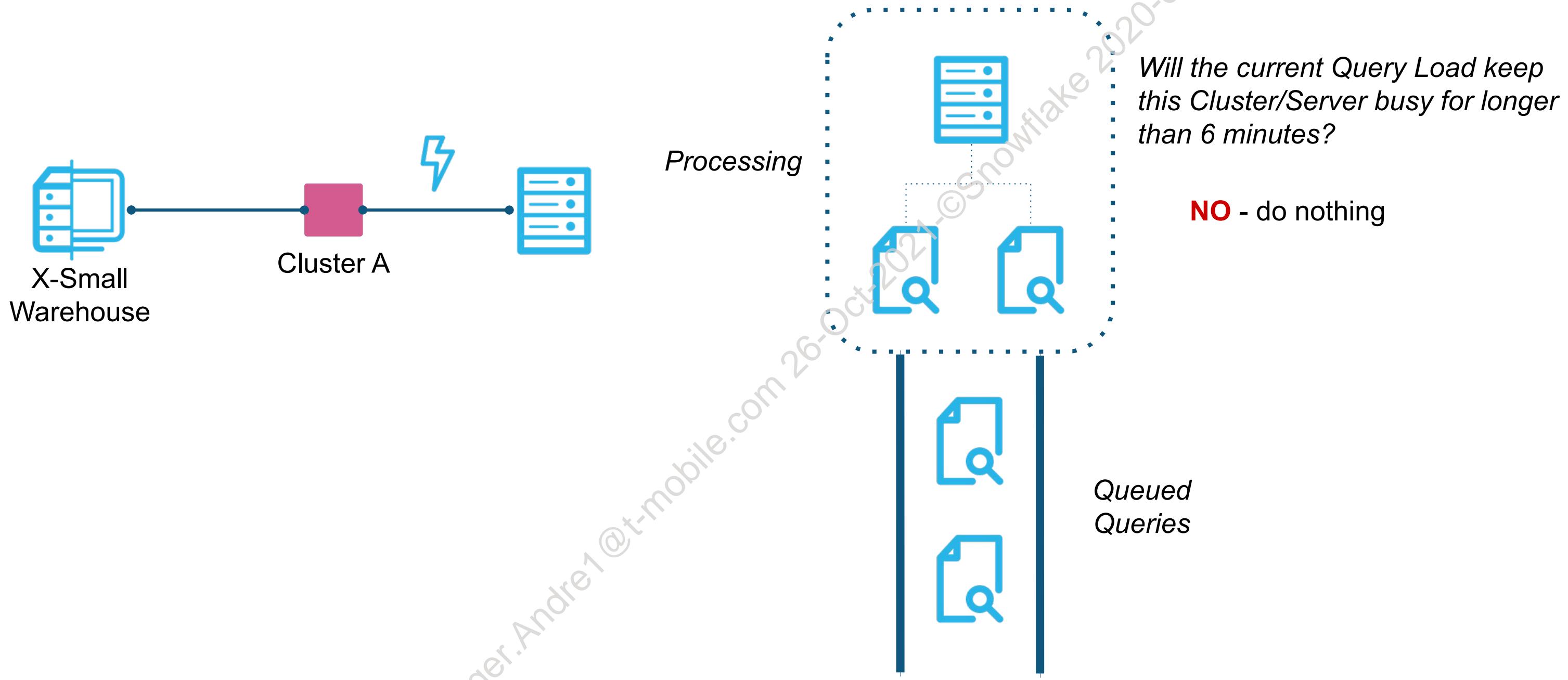
SCALING OUT POLICY: ECONOMY



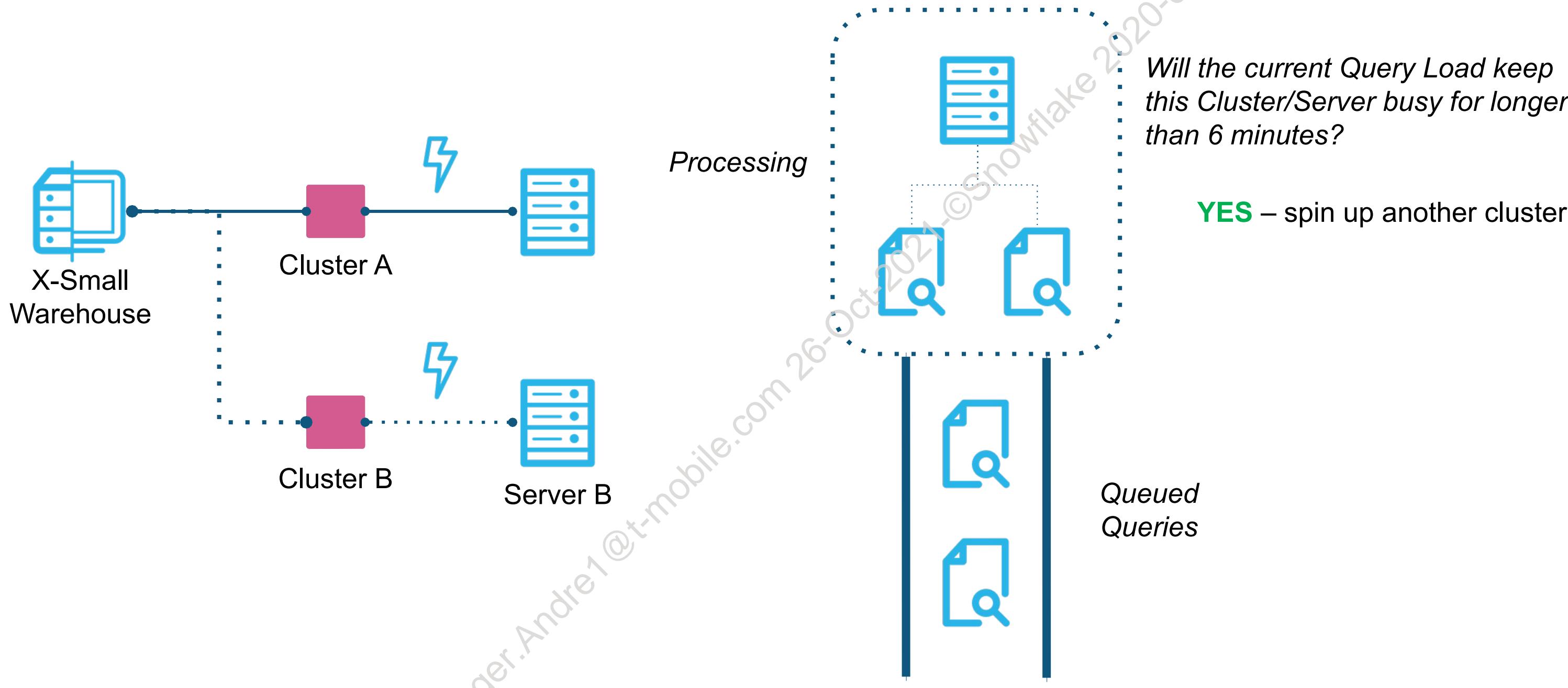
SCALING OUT POLICY: ECONOMY



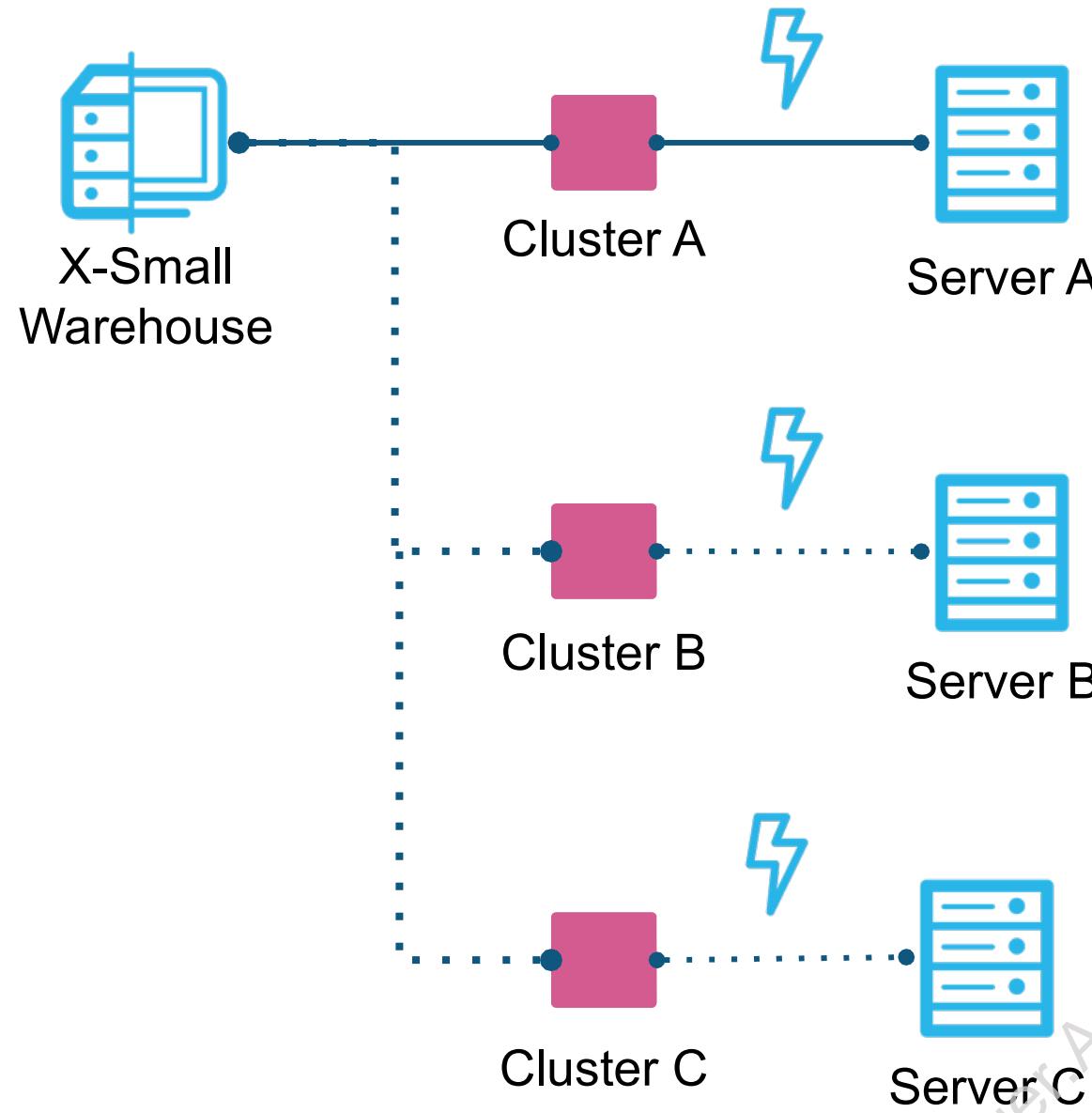
SCALING OUT POLICY: ECONOMY



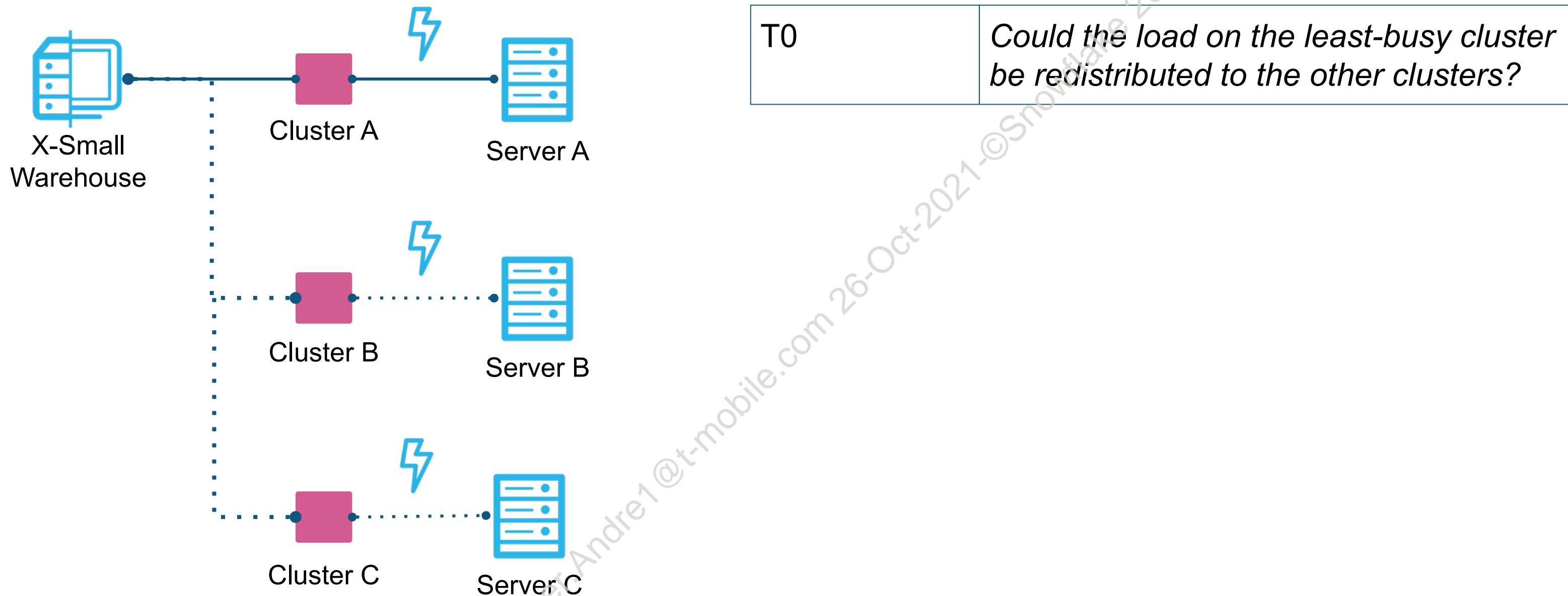
SCALING OUT POLICY: ECONOMY



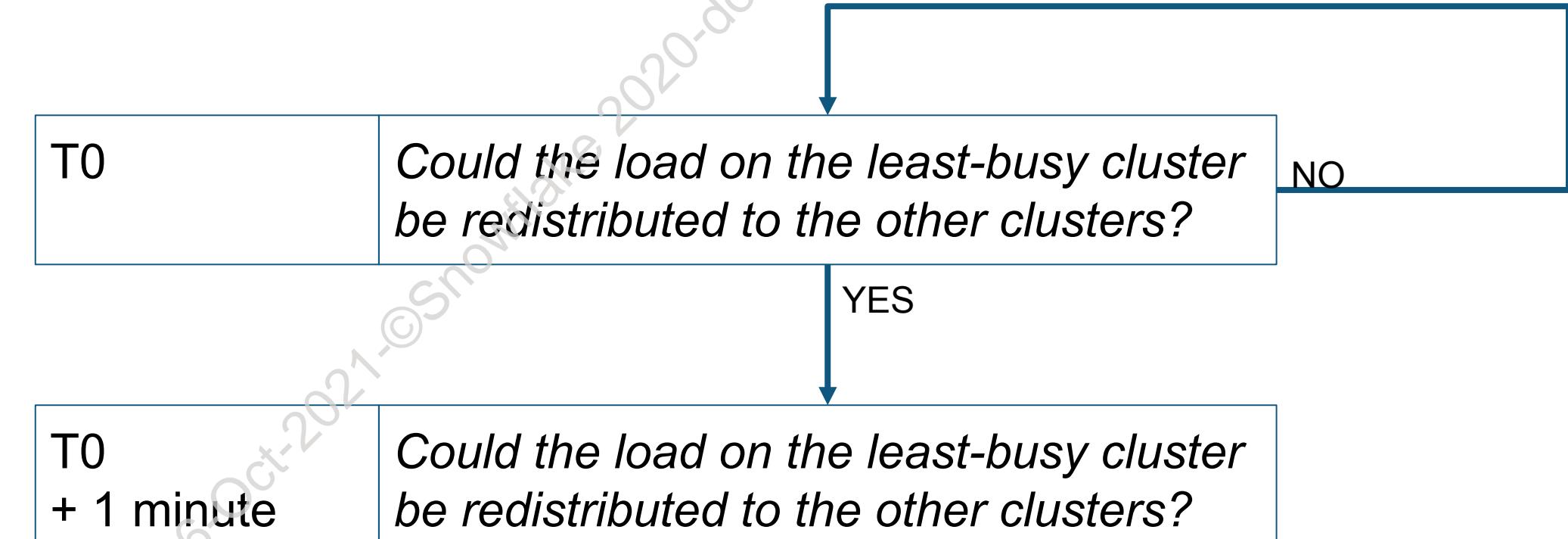
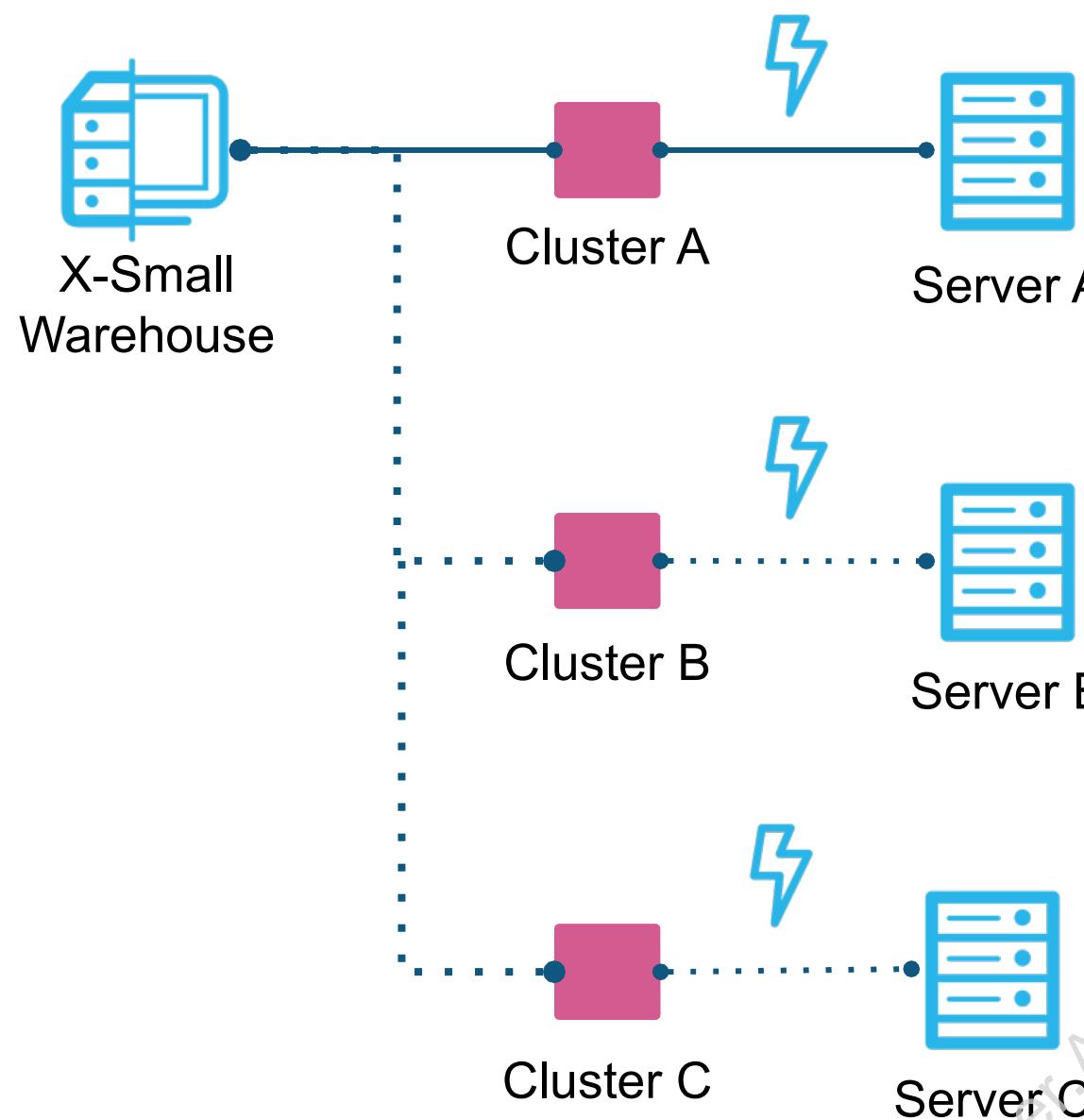
SCALING BACK POLICY: STANDARD



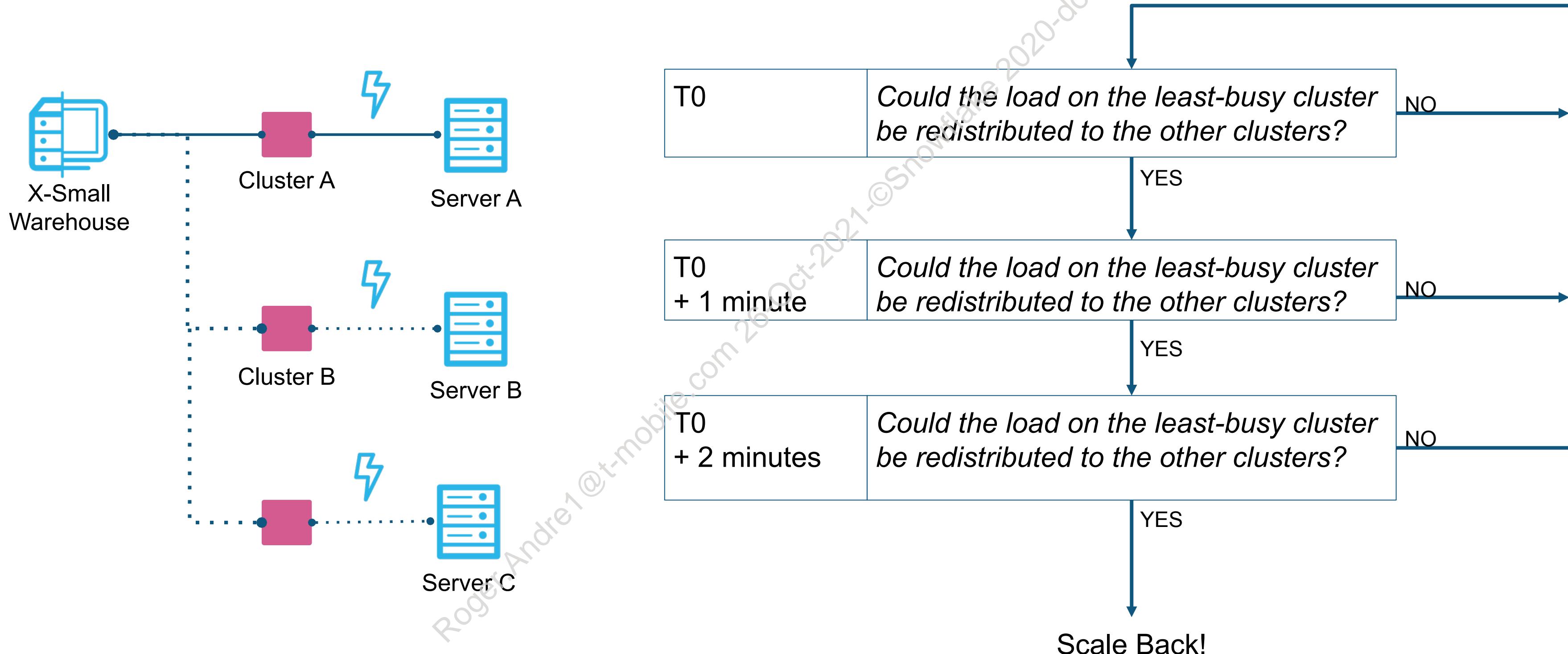
SCALING BACK POLICY: STANDARD



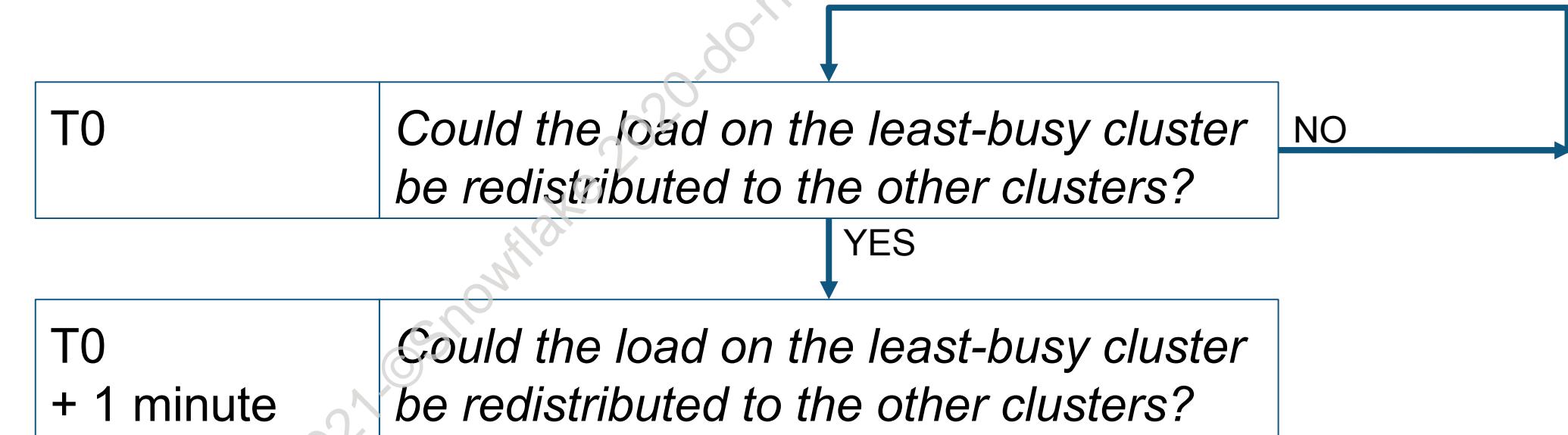
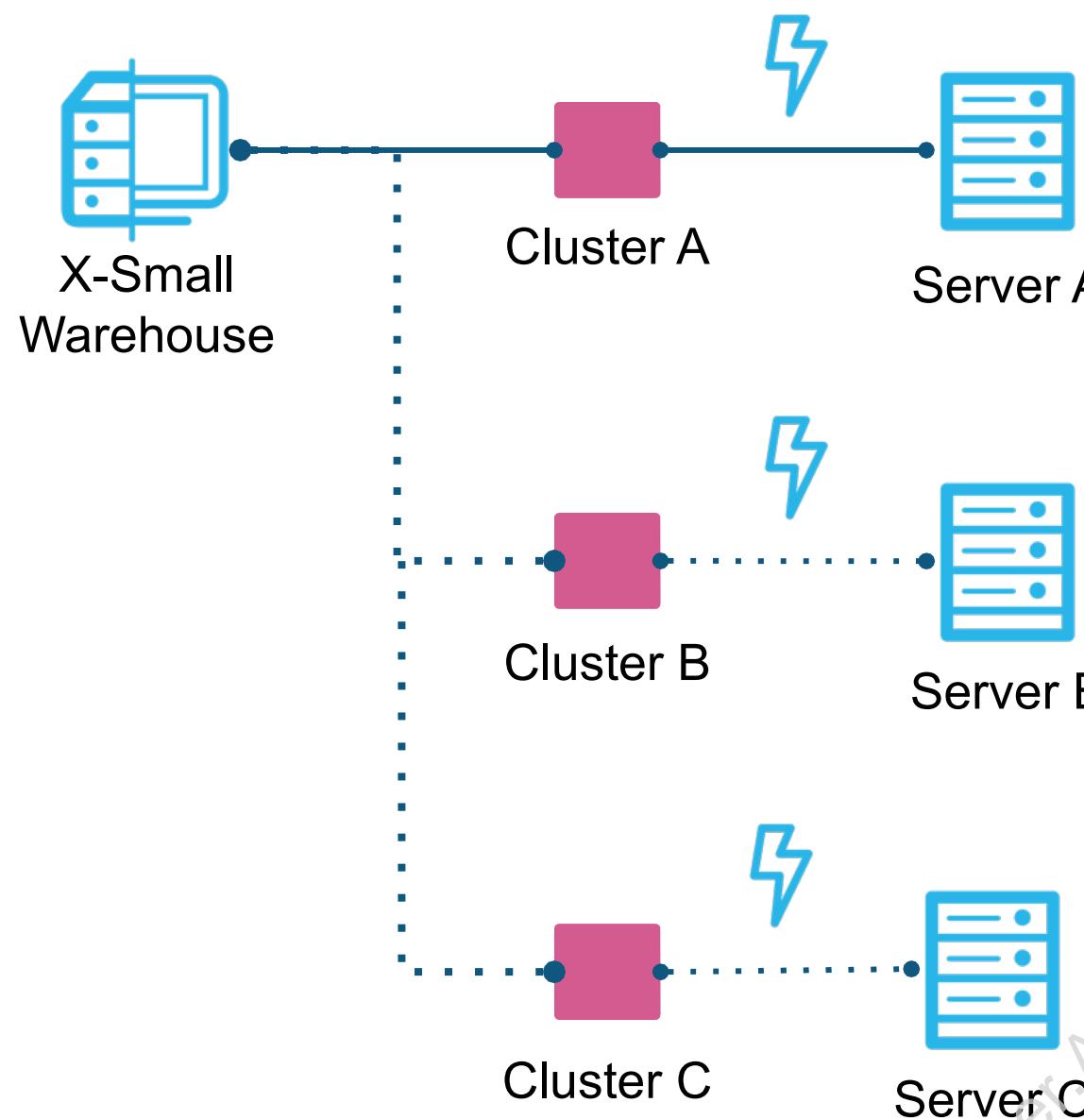
SCALING BACK POLICY: STANDARD



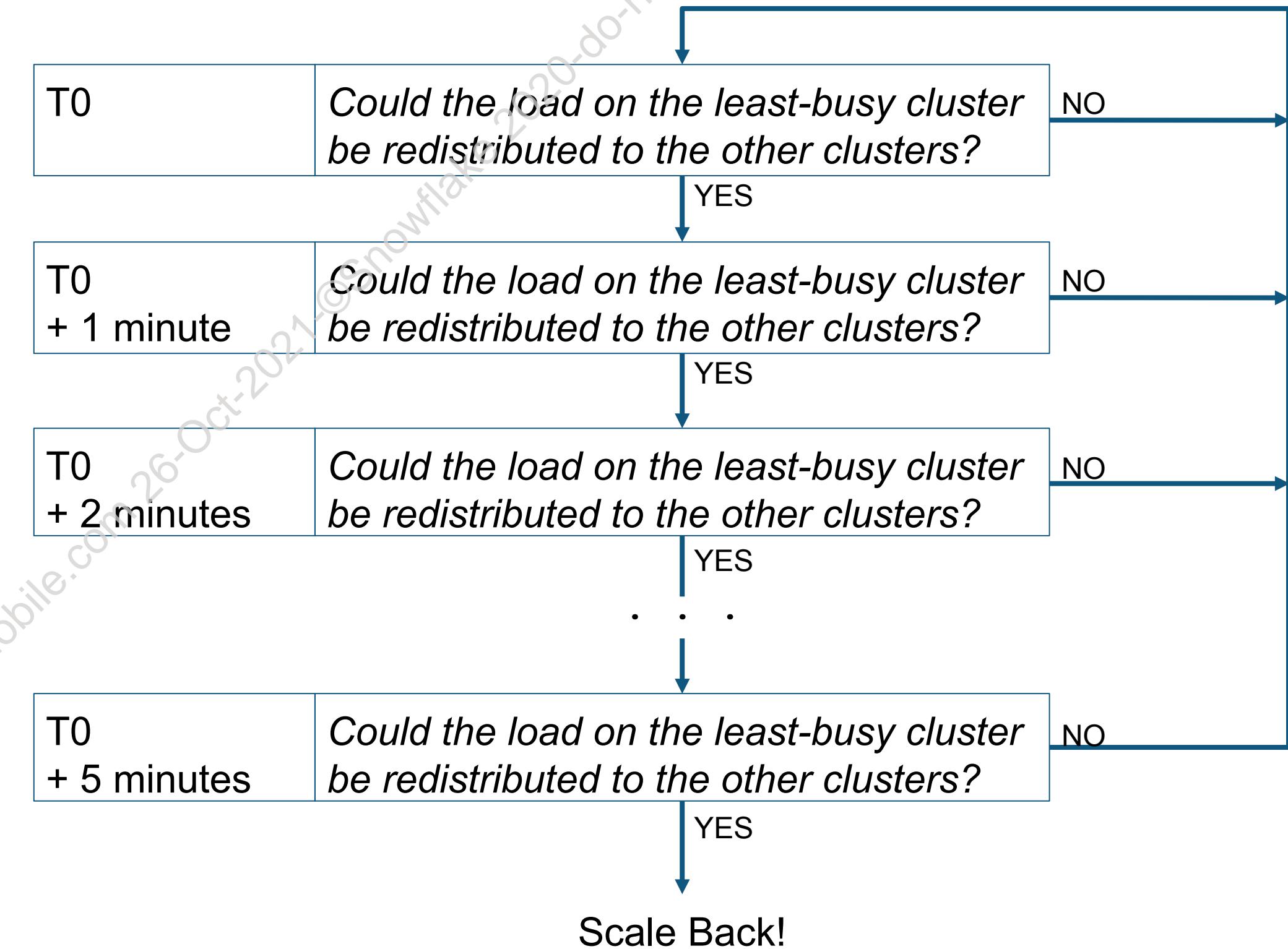
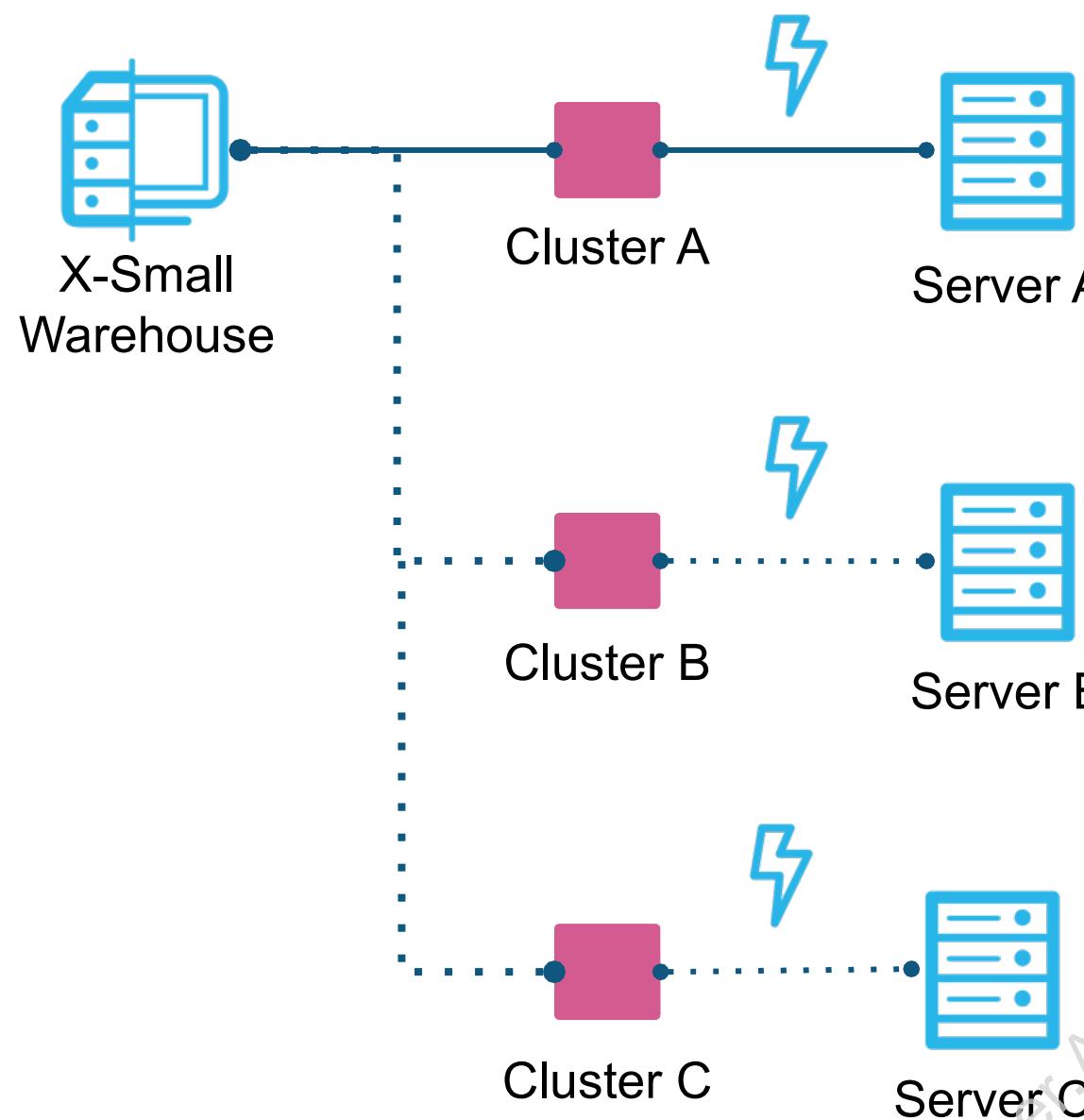
SCALING BACK POLICY: STANDARD



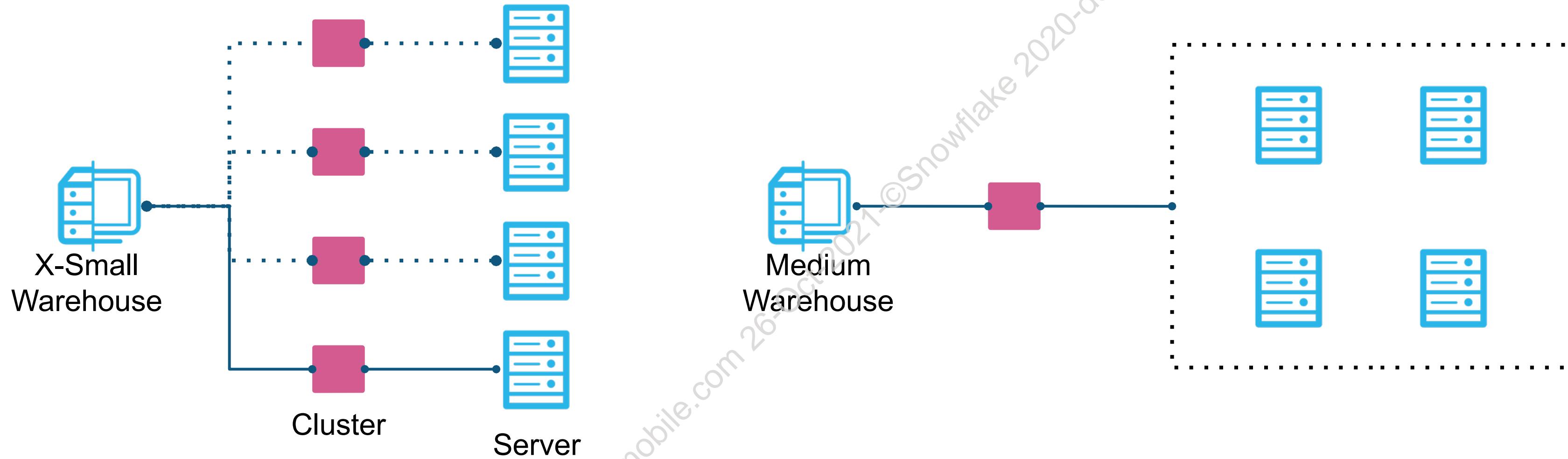
SCALING BACK POLICY: ECONOMY



SCALING BACK POLICY: ECONOMY



ARE THESE WAREHOUSES EQUIVALENT?



Auto-Scale, Multi-Cluster X-Small Warehouse

- Minimum of 1
- Maximum of 4

Medium Warehouse

- One cluster of 4 servers



ALTERING QUERY BEHAVIOR

STATEMENT_QUEUED_TIMEOUT_IN_SECONDS

- How long a queued query will wait before being cancelled by the system
- Can be set for Account/Session/User or Virtual Warehouse
- Default: 0

STATEMENT_TIMEOUT_IN_SECONDS

- How long a query can run before being cancelled by the system
- Set for Account/Session/User or Virtual Warehouse
- Default: 2 days → Deployment best practice → lower at the account level to your preference



LAB EXERCISE: 10

Determine Appropriate Warehouse Sizes

30 minutes

Tasks:

- JOIN using an XSmall warehouse
- JOIN using a Medium warehouse

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



LAB RECAP

- Below are the results from running the same query against various warehouses sizes.
- Which warehouse size would you recommend, and why?

Warehouse Size	Query Time
Small	24m 30s
Medium	10m 00s
Large	5m 00s
XLarge	2m 40s
XXLarge	1m 40s



LAB RECAP

- Below are the results from running the same query against various warehouses sizes.
- Which warehouse size would you recommend, and why?

Warehouse Size	Query Time	Credits
Small	24m 30s	.882
Medium	10m 00s	.660
Large	5m 00s	.660
XLarge	2m 40s	.704
XXLarge	1m 40s	.890



ACCOUNT & RESOURCE MANAGEMENT

Roger.Andre1@t-mobile.com 26-Oct-2021 ©
Snowflake 2020-do-not-copy



MODULE AGENDA

- Controlling Costs
- Resource Monitors
- INFORMATION_SCHEMA
- SNOWFLAKE Database

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



CONTROLLING COSTS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



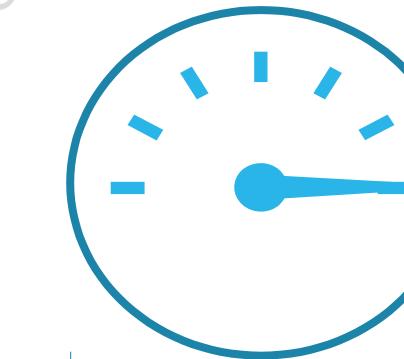
PERFORMANCE AND USAGE CONSIDERATIONS

COST



- Storage
- Virtual warehouse compute
- Cloud services compute
- Data transfer/egress charges

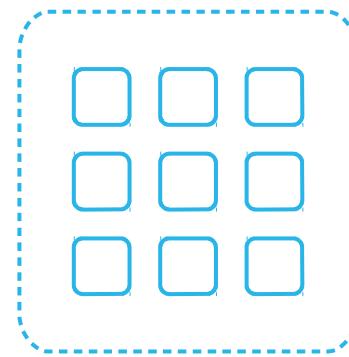
PERFORMANCE



- Query workload
- Query complexity
- Warehouse size
- SQL efficiency



STORAGE USAGE



Database Tables

(Active, Time Travel, Failsafe)



Internal Stages

(User, Table, Named)

- Billed monthly
 - Calculated based on daily bytes of all data stored each day in account
- Cost varies based on type of account, cloud provider, and region
- Storage costs include:
 - Active storage – tables and objects
 - Time travel storage
 - Failsafe storage
 - Internal stages



SAVE ON STORAGE COSTS

- Remove transient tables/schemas/databases when they are no longer needed
- Remove files in internal stages
 - Or use the PURGE option with COPY INTO
- Use appropriate time travel settings
 - Set at the account, database, schema, or table level
- Capacity pricing is less expensive than pay-as-you-use



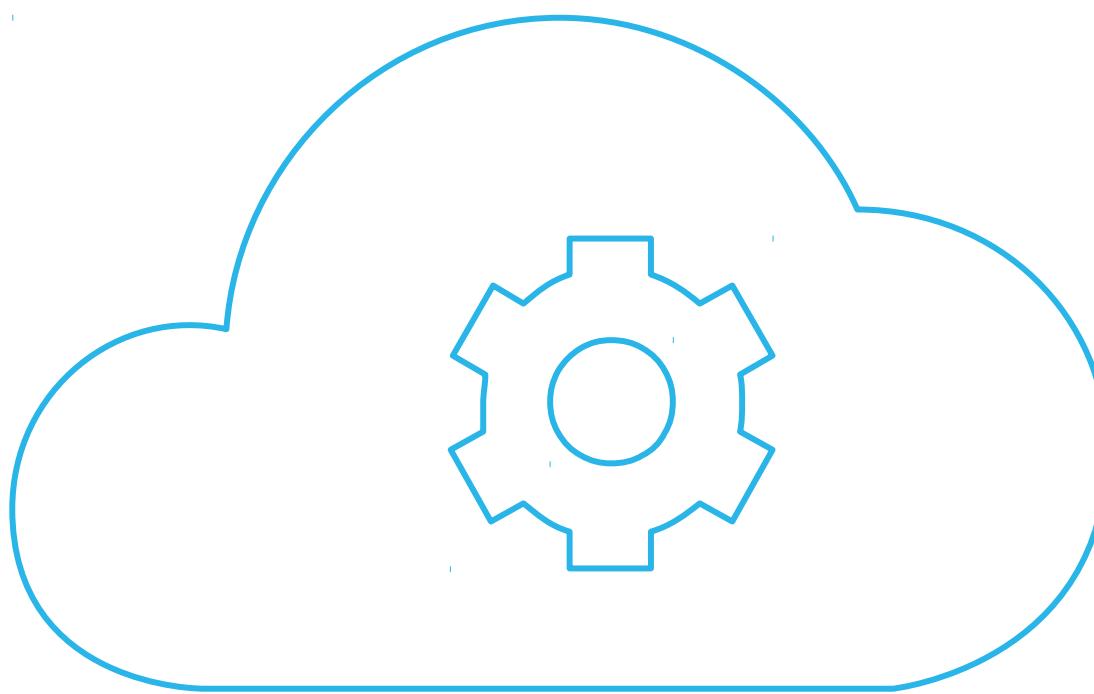
USER-MANAGED COMPUTE

Warehouse Size	Servers / Cluster	Credits / Hour	Credits / Second
X-Small	1	1	0.0003
Small	2	2	0.0006
Medium	4	4	0.0011
Large	8	8	0.0022
X-Large	16	16	0.0044
2X-Large	32	32	0.0089
3X-Large	64	64	0.0178
4X-Large	128	128	0.0356

- Only billed when running
- Billed per-second, with a 60-second minimum
- Can be limited and controlled via resource monitors
- Usage can be viewed via the UI or Information Schema/Account Usage



SERVERLESS FEATURES and CLOUD COMPUTE



- Billed per second
- No virtual warehouses needed
- Includes:
 - Serverless Features Credits
 - Snowpipe
 - Materialized view maintenance
 - Clustered table maintenance
 - Cloud Services over 10% of the compute total
 - Metadata queries
 - Query result cache use

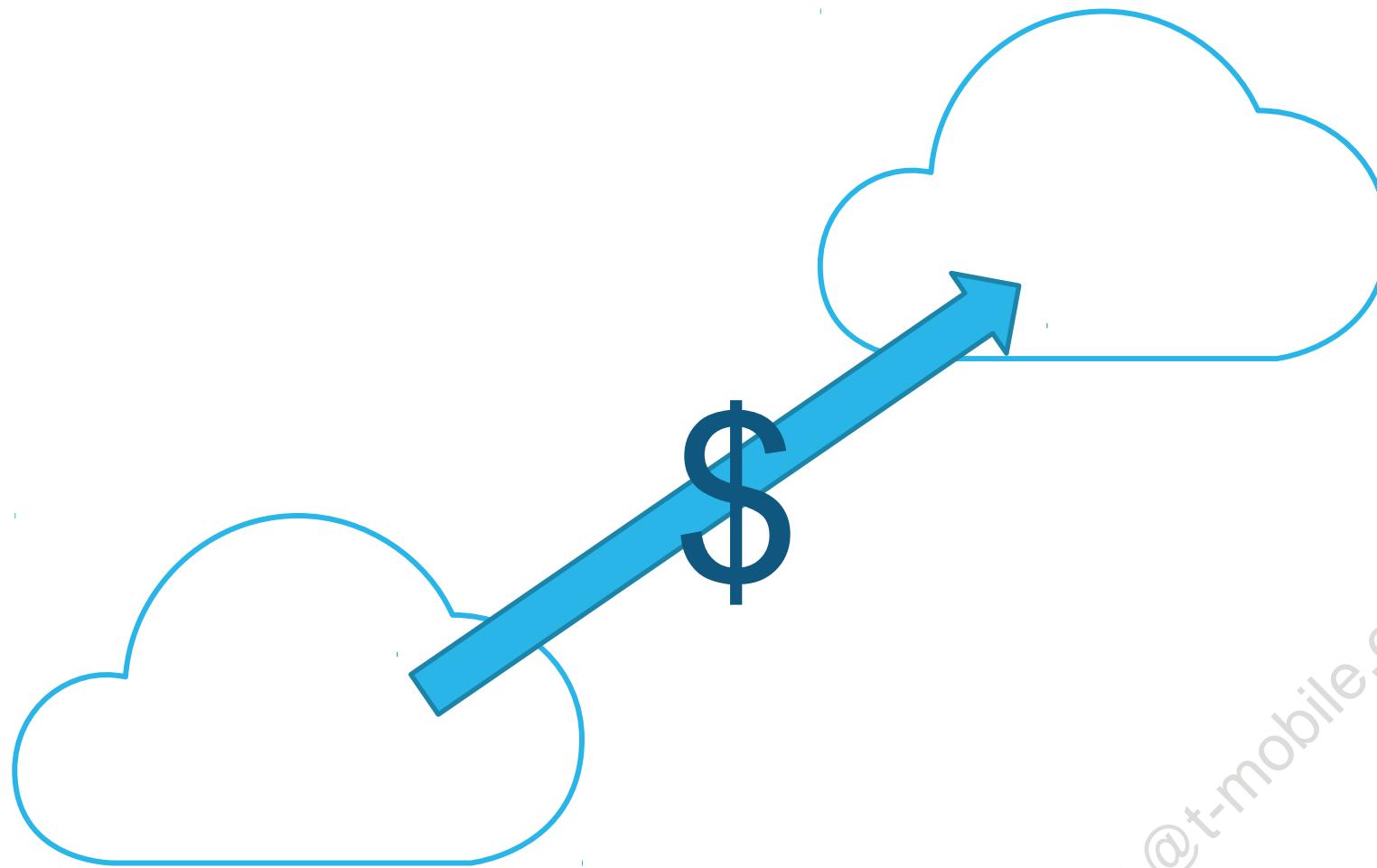


CONTROL COMPUTE COSTS

- Right-size your virtual warehouses
- Consider data cache when setting auto-suspend values
- Use multi-cluster warehouses and auto-scaling
- Limit who can create or manage warehouses
- Create resource monitors to track consumption
 - Don't forget reader accounts!
- Track and manage cloud compute costs



DATA TRANSFER CHARGES



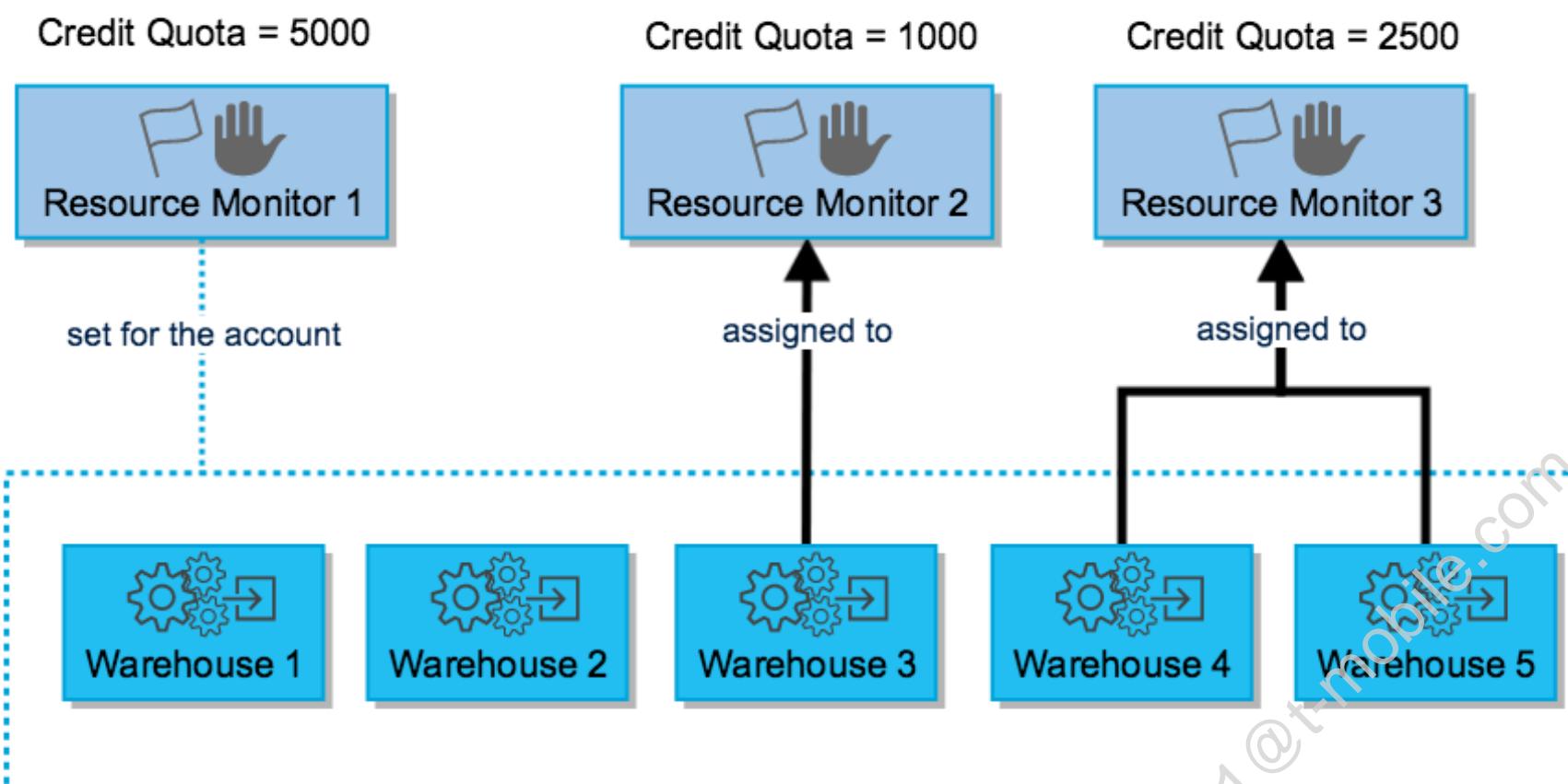
- Database replication
- Varies based on source and destination
 - Region to region transfer within a cloud provider are less expensive than transfers from one cloud provider to another
- Charges for:
 - Initial replication
 - Refreshes

RESOURCE MONITORS

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



RESOURCE MONITORS



- Help control costs and avoid unexpected warehouse credit usage.
- Tracks warehouse use and cloud computing.
- Set at the account or warehouse level
- Can trigger various actions
 - Sending alert notifications
 - Suspending the warehouse(s)



RESOURCE MONITORS - DETAILS

- ACCOUNTADMIN privileges are required to set up Resource Monitors
- Resource Monitor attributes:
 - Credit quota
 - Level (account, warehouse, or warehouses)
 - Schedule (start, end, frequency)
 - Action (notify, suspend, suspend immediate)
- Not intended to strictly control usage to the second
 - Quota may be exceeded before action is triggered
- **Must enable notifications in Preferences**



CREATE RESOURCE MONITORS

- Through the UI
- With SQL

```
CREATE RESOURCE MONITOR  
ALTER RESOURCE MONITOR  
SHOW RESOURCE MONITORS  
DROP RESOURCE MONITOR
```

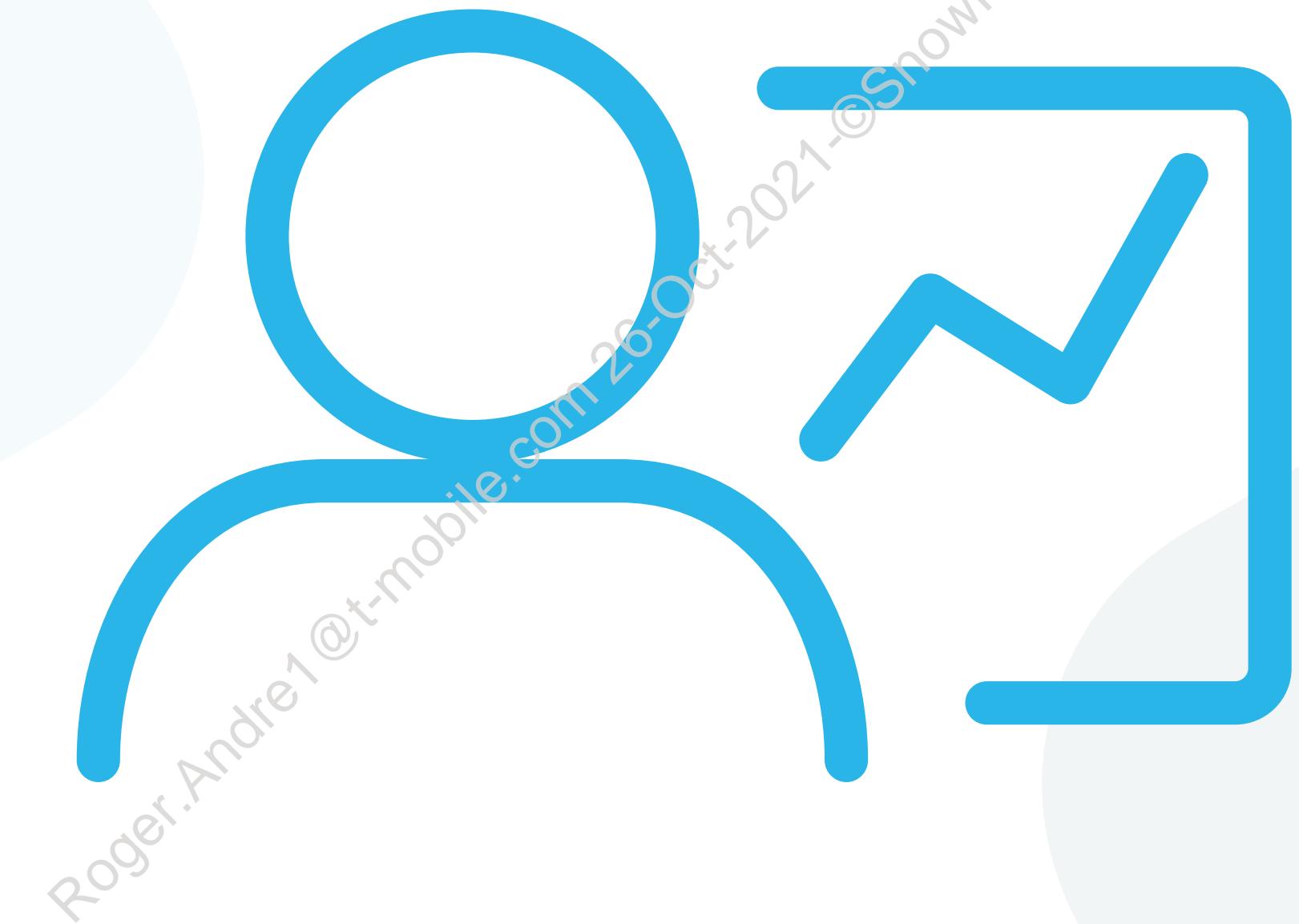
Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



INSTRUCTOR DEMO

Set Up Resource Monitors

10 minutes

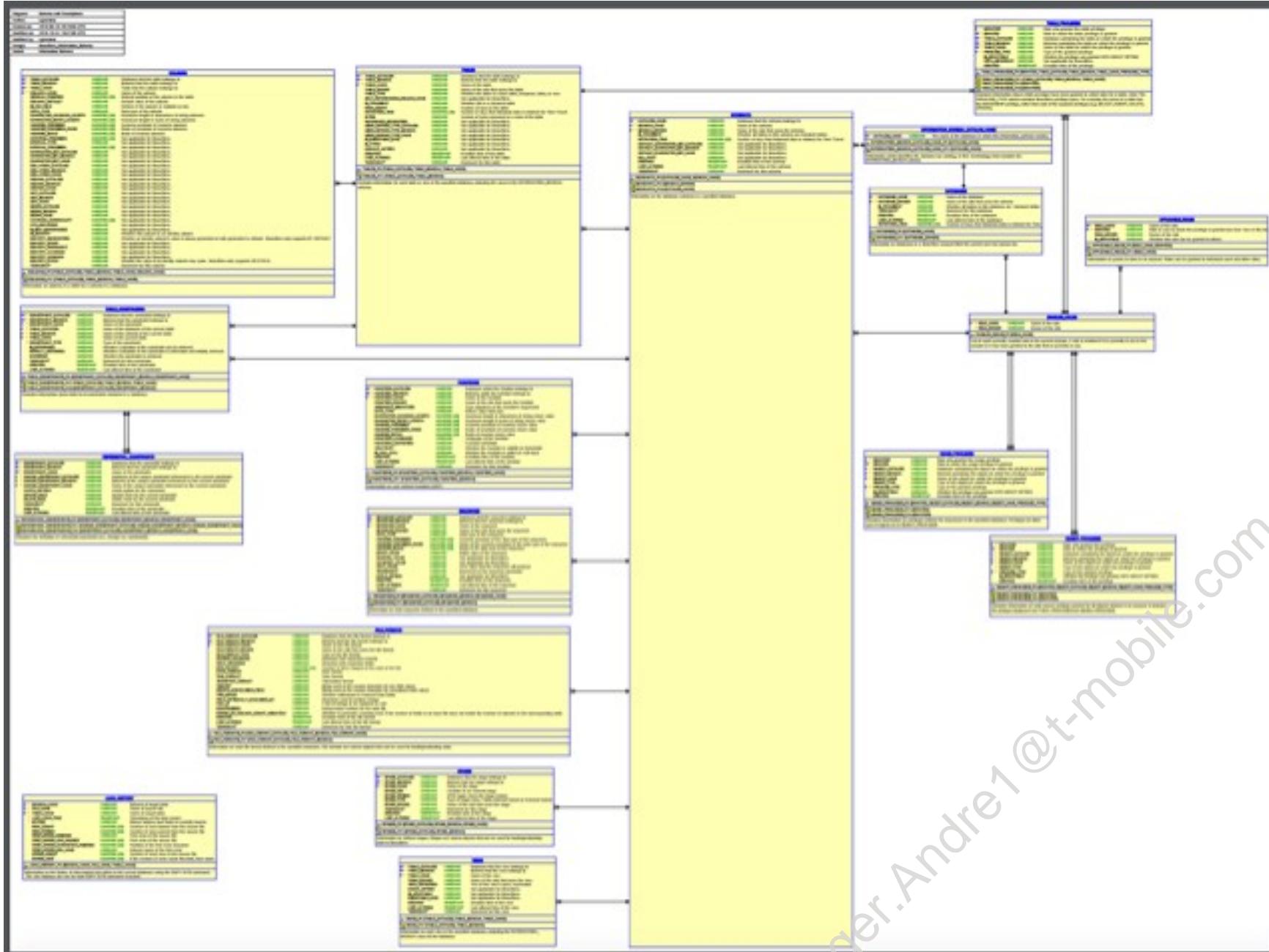


INFORMATION_SCHEMA

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



INFORMATION_SCHEMA

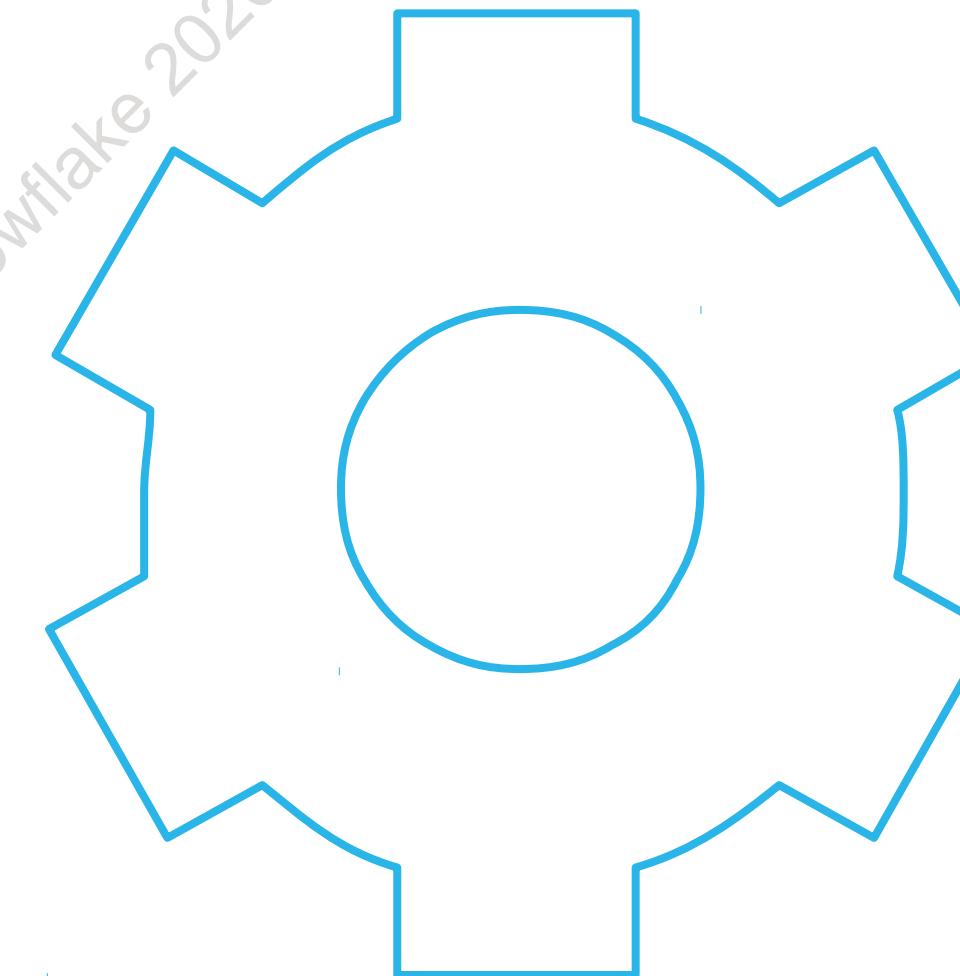


- Set of system-defined views and table functions that provide metadata information about objects
- Based on the SQL-92 ANSI Information Schema + additional views and functions specific to Snowflake
- Queries return only objects to which the current role has been granted access



INFORMATION_SCHEMA

- Built-in, read-only schema
- Exists for each database
- System-defined views and table functions
 - Views for all objects in the database
 - Views for account-level objects
 - Table functions for historical and usage data
- Provide extensive “logging” metadata information



SOME VIEWS IN INFORMATION_SCHEMA

- APPLICABLE_ROLES
- COLUMNS
- DATABASES
- ENABLED_ROLES
- FILE_FORMATS
- FUNCTIONS
- LOAD_HISTORY
- PIPES
- PROCEDURES
- SEQUENCES
- STAGES
- TABLE_PRIVILEGES
- TABLE_STORAGE_METRICS
- TABLES
- USAGE_PRIVILEGES
- VIEWS



TABLE FUNCTIONS IN INFORMATION_SCHEMA

See documentation for a full list

- AUTOMATIC_CLUSTERING_HISTORY
- DATABASE_STORAGE_USAGE_HISTORY
- LOGIN_HISTORY
- LOGIN_HISTORY_BY_USER
- MATERIALIZED_VIEW_REFRESH_HISTORY
- QUERY_HISTORY
- WAREHOUSE_LOAD_HISTORY



INFORMATION_SCHEMA EXAMPLES

List object privileges

```
SELECT object_type, object_name, privilege_type, grantor, grantee  
FROM INFORMATION_SCHEMA.OBJECT_PRIVILEGES  
ORDER BY object_type, object_name;
```

OBJECT_TYPE	OBJECT_NAME	PRIVILEGE_TYPE	GRANTOR	GRANTEE
DATABASE	DBHOL	OWNERSHIP	SYSADMIN	SYSADMIN
DATABASE	MY_DB	OWNERSHIP	TRAINING_ROLE	TRAINING_ROLE
DATABASE	SNOWFLAKE	USAGE	ACCOUNTADMIN	PUBLIC
SCHEMA	PUBLIC	OWNERSHIP	TRAINING_ROLE	TRAINING_ROLE
TABLE	TEST	OWNERSHIP	TRAINING_ROLE	TRAINING_ROLE



INFORMATION_SCHEMA EXAMPLES

List total table size, by table, in megabytes (MB)

```
SELECT table_schema, table_name, table_owner, table_type,  
       ROUND(bytes/1024/1024,3) AS mb  
FROM INFORMATION_SCHEMA.TABLES  
WHERE TABLE_SCHEMA NOT IN ('INFORMATION_SCHEMA');
```

TABLE_SCHEMA	TABLE_NAME	TABLE_OWNER	TABLE_TYPE	MB
PUBLIC	ORDERS	TRAINING_ROLE	BASE TABLE	2043.877
PUBLIC	CUSTOMER	TRAINING_ROLE	BASE TABLE	103.143
DEV	MY_TEMP_TABLE	DBA	LOCAL TEMPORARY	1048.304



SNOWFLAKE DATABASE

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



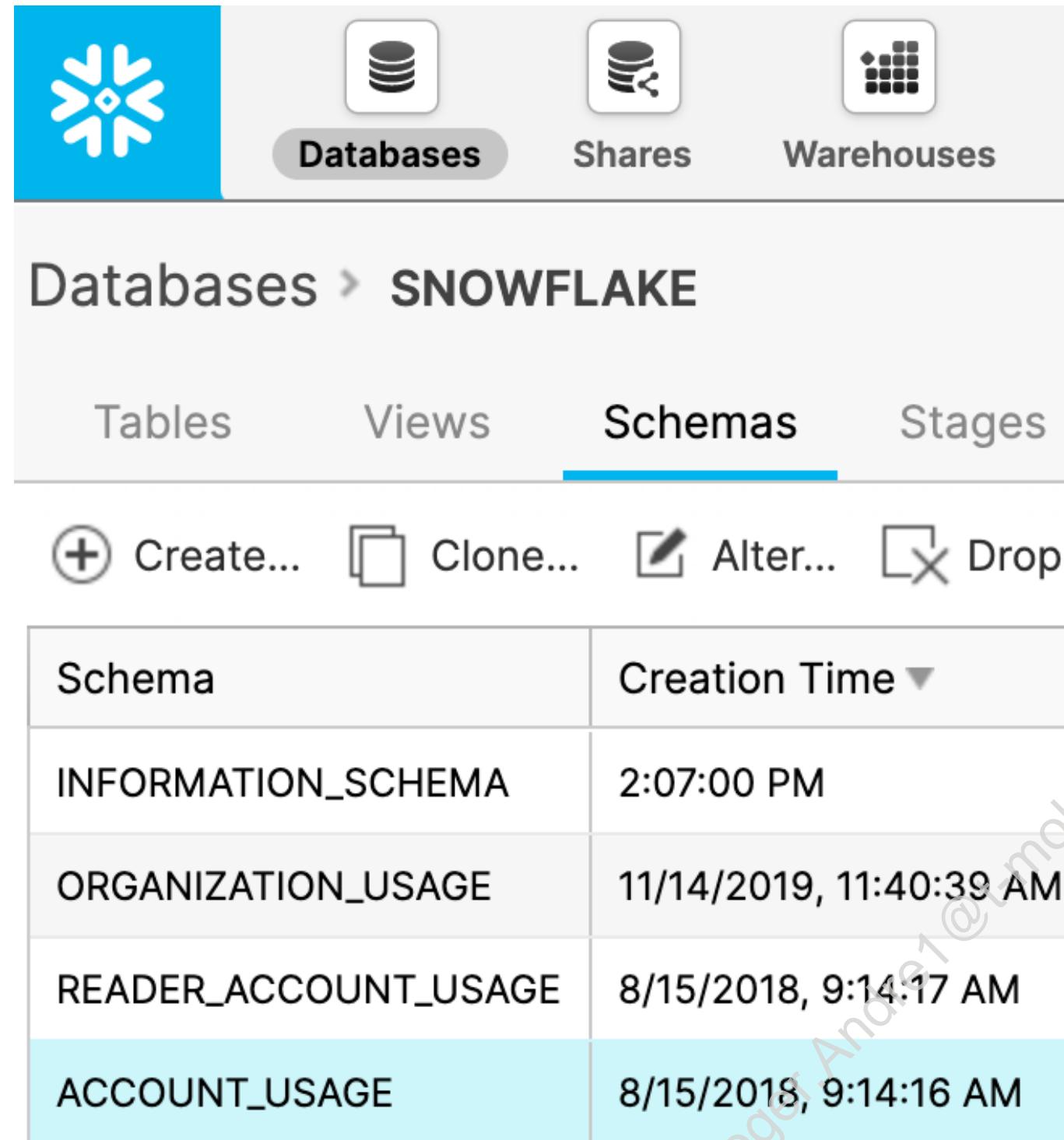
SNOWFLAKE DATABASE

- Shared by Snowflake as a collection of secured views.
- By default, only ACCOUNTADMIN can access the SNOWFLAKE database and schemas
- Privileges can be granted to other roles

```
GRANT IMPORTED PRIVILEGES ON DATABASE SNOWFLAKE TO ROLE <role>;
```



SNOWFLAKE DATABASE



The screenshot shows the Snowflake web interface for managing databases. At the top, there are icons for Databases, Shares, and Warehouses. Below this, the path 'Databases > SNOWFLAKE' is displayed. A navigation bar includes 'Tables', 'Views', 'Schemas' (which is underlined in blue), and 'Stages'. Below the navigation bar are buttons for 'Create...', 'Clone...', 'Alter...', and 'Drop...'. The main area displays a table of schemas:

Schema	Creation Time
INFORMATION_SCHEMA	2:07:00 PM
ORGANIZATION_USAGE	11/14/2019, 11:40:39 AM
READER_ACCOUNT_USAGE	8/15/2018, 9:14:17 AM
ACCOUNT_USAGE	8/15/2018, 9:14:16 AM

- Shared by Snowflake to all accounts
 - Example of secure data sharing
- Contains the following schemas:
 - ACCOUNT_USAGE
 - INFORMATION_SCHEMA
 - READER_ACCOUNT_USAGE
 - ORGANIZATION_USAGE
- Contains historical usage data and logs specific to your account



ACCOUNT_USAGE

The screenshot shows the Snowflake web interface. At the top, there's a blue header bar with the Snowflake logo and the word "snowflake". Below it, a navigation bar has "Databases" selected, showing icons for Databases and Shares. The main content area shows "Databases > SNOWFLAKE". Under "Views", there are tabs for "Tables", "Views" (which is underlined), "Schemas", and "Stages". Below these are buttons for "Create...", "Drop...", and "Transfer Ownership". A table lists six views with their respective schemas:

View Name	Schema
FUNCTIONS	ACCOUNT_USAGE
LOGIN_HISTORY	ACCOUNT_USAGE
LOAD_HISTORY	ACCOUNT_USAGE
COPY_HISTORY	ACCOUNT_USAGE
WAREHOUSE_LOAD_HISTORY	ACCOUNT_USAGE

- Contains views that display object metadata and usage metrics
- Generally, ACCOUNT_USAGE mirrors the corresponding views and table functions in INFORMATION_SCHEMA, with some differences



READER_ACCOUNT_USAGE

The screenshot shows the Snowflake web interface. In the top navigation bar, the 'snowflake' logo is on the left, followed by three icons: 'Databases' (selected), 'Shares', and 'Data Exchange'. Below the navigation bar, the path 'Databases > SNOWFLAKE' is displayed. Underneath this, there are tabs for 'Tables', 'Views' (which is underlined in blue), 'Schemas', 'Stages', and 'File Format'. Below the tabs are buttons for 'Create...', 'Drop...', and 'Transfer Ownership'. The main content area displays a table with the following data:

View Name	Schema ▾
LOGIN_HISTORY	READER_ACCOUNT_USAGE
QUERY_HISTORY	READER_ACCOUNT_USAGE
RESOURCE_MONITORS	READER_ACCOUNT_USAGE
STORAGE_USAGE	READER_ACCOUNT_USAGE
WAREHOUSE_METERING_HIS...	READER_ACCOUNT_USAGE

- Contains views that apply to reader accounts
- Small subset of views in ACCOUNT_USAGE



INFORMATION_SCHEMA

The screenshot shows the Snowflake web interface. At the top, there's a blue header bar with the Snowflake logo and navigation links for Databases, Shares, and Data Exchange. Below the header, the path 'Databases > SNOWFLAKE' is displayed. A navigation bar contains tabs for Tables, Views (which is underlined in blue), Schemas, Stages, and File Format. Below this, there are buttons for Create..., Drop..., and Transfer Ownership. The main content area displays a table with the following data:

View Name	Schema ▾
APPLICABLE_ROLES	INFORMATION_SCHEMA
COLUMNS	INFORMATION_SCHEMA
DATABASES	INFORMATION_SCHEMA
ENABLED_ROLES	INFORMATION_SCHEMA
EXTERNAL_TABLES	INFORMATION_SCHEMA

- Schema automatically created in all databases
- Does not serve a purpose in shared databases, and can be disregarded



ACCOUNT_USAGE VS. INFORMATION_SCHEMA

ACCOUNT_USAGE (IN SNOWFLAKE DATABASE)	INFORMATION_SCHEMA (IN INDIVIDUAL DATABASES)
Includes dropped objects	Does not include dropped objects
45 minutes to 3 hours latency (varies by view)	No latency
Data retained for 1 year	Data retained up to 6 months (varies by view/table function)



ACCOUNT_USAGE EXAMPLES

Login History for a Specific User Over the Last Week:

```
SELECT event_id, event_timestamp, event_type, user_name, error_code, error_message
FROM SNOWFLAKE.ACCOUNT_USAGE.LOGIN_HISTORY
WHERE user_name = 'Anne_Jordan'
    AND EVENT_TIMESTAMP >= DATEADD('DAYS', -8, CURRENT_DATE())
    AND EVENT_TIMESTAMP < CURRENT_DATE()
ORDER BY EVENT_TIMESTAMP;
```

EVENT_ID	EVENT_TIMESTAMP	EVENT_TYPE	USER_NAME	ERROR_CODE	ERROR_MESSAGE
124077310283306	2019-09-05 10:58:39.777 -0700	LOGIN	ANNE_JORDAN	390100	INCORRECT_USE...
124077310289407	2019-09-12 08:21:43.103 -0700	LOGIN	ANNE_JORDAN	NULL	NULL



ACCOUNT_USAGE EXAMPLES

Query History for a Specific Warehouse Over the Last Month:

```
SELECT *
FROM SNOWFLAKE.ACCOUNT_USAGE.QUERY_HISTORY
WHERE WAREHOUSE_NAME = 'DEMO_WH'
    AND START_TIME >= DATEADD('DAYS', -31, CURRENT_DATE())
    AND START_TIME < CURRENT_DATE()
ORDER BY START_TIME;
```

QUERY_ID	QUERY_TEXT	DATABASE_ID	DATABASE_NAME	SCHEMA_ID
018eb0f9-0...	SHOW WAREHOUSES;	NULL	NULL	NULL
018eb0fa-0...	GRANT CREATE TABL...	239	FINANCE	NULL



LAB EXERCISE: 11

Monitor Billing & Usage Information

40 minutes

Tasks:

- View warehouse usage
- View billing and usage
- Monitor storage
- Query INFORMATION_SCHEMA and ACCOUNT_USAGE



Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy

PREVIEW FEATURES

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



MODULE AGENDA

- Overview
- Releases
- Snowsight
- Public Preview Features
- Private Preview Features
- Roadmap

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



OVERVIEW

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



WHAT IS A PREVIEW FEATURE?

- Implemented and tested in Snowflake
- Full usability and corner-case handling may not be complete
- Use of preview features are not warranted against defects that may produce undesired results
 - Behavior may change between Preview and Release
- Not recommended for production use



BEST PRACTICES FOR PREVIEW FEATURES

- Do not use on production data
- Do not use on production data
- Do not use on production data
- Contact support for additional information if you really want to use it on production data



PREVIEW TYPES

- Private Preview / On Request
 - Typically invite-only at first
 - Generally, in the early stages of preview
 - Work with your account team if access is required
- Public Preview / Open
 - Enabled by default for all accounts
 - Free to use, though still carry some risk and are not for full production use



RELEASES

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



RELEASE SCHEDULE

- Snowflake generally releases weekly*, with releases typically completed on Fridays
- In the case of bug fixes/hotfixes, releases will be rolled out as needed
- Releases are seamless to the Customer--they require no downtime, but take effect as soon as they are committed

*** There are a few weeks of the year--typically around the end-of-year holidays--in which releases will not be rolled out**



RELEASE NOTES

- Snowflake publishes release notes detailing new functionality and behavior changes to existing functionality; these notes are available via two different locations and formats depending on your needs:
 - Do you want the detail behind each individual release and behavior change? See the [Lodge Community Announcements](#)
 - Do you want a summary, by month, of new features and behavior changes? See the [Documentation Release Notes](#)



PENDING BEHAVIOR CHANGES

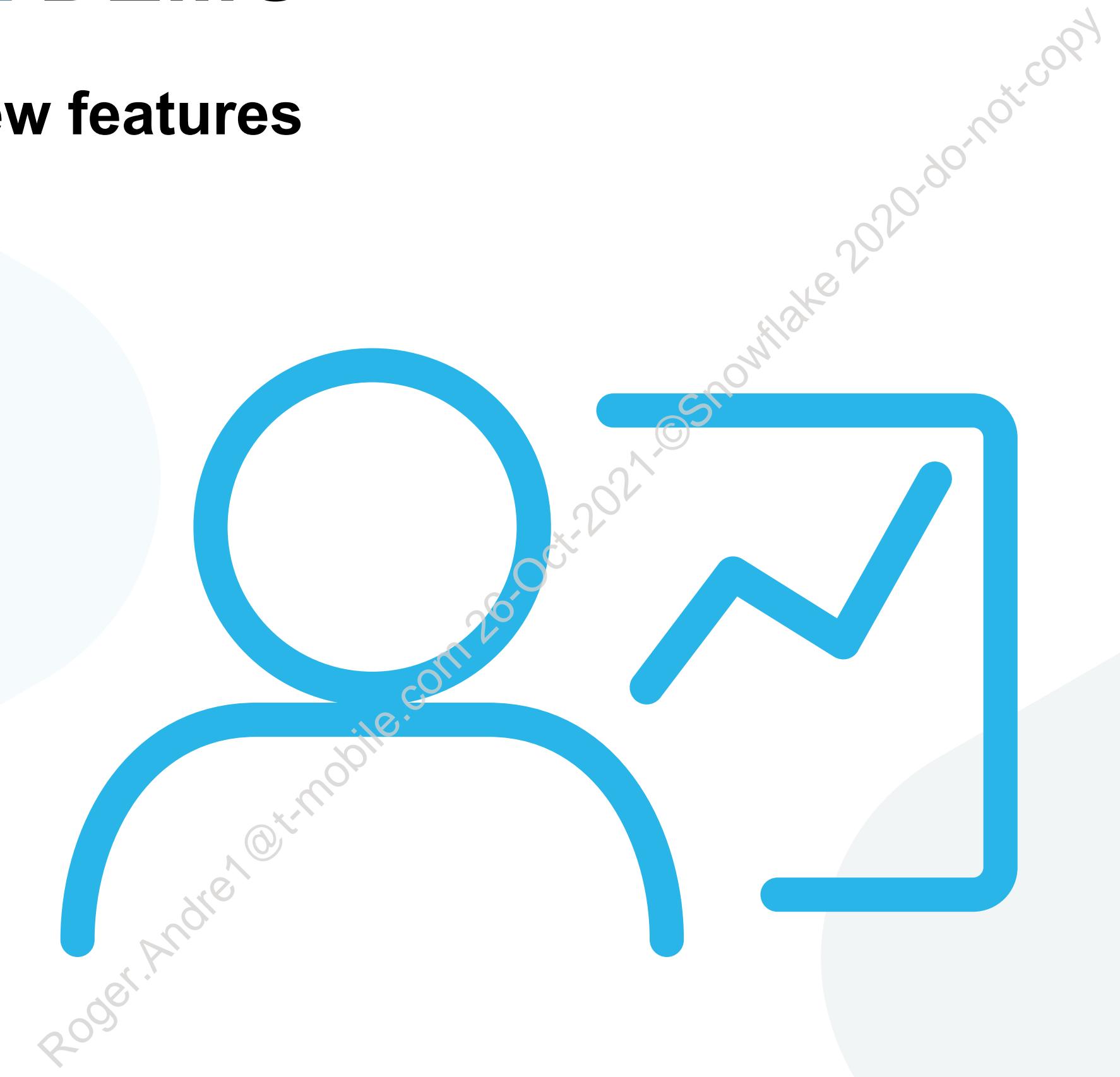
- Snowflake publishes upcoming behavior changes in advance of their release so you may adequately prepare
- The Pending Behavior Change Log will detail the upcoming change and its release date:
<https://support.snowflake.net/s/article/Pending-Behavior-Change-Log>
- This Log will also detail recently released behavior changes



INSTRUCTOR DEMO

Private/public preview features

15 minutes



LAB EXERCISE: 12

Complete the Course Survey

10 minutes

- Please!
- Take a few minutes before we move on to the last section to fill out your course surveys



FUNDAMENTALS RECAP

Roger.Andre1@t-mobile.com 26 Oct 2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Can you have two tables with the same name in your account?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: Can you have two tables with the same name in your account?

A: Yes, as long as the full path (*database.schema.table*) is different



QUESTION

Q: What are the elements of a worksheet context?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What are the elements of a worksheet context?

A: Role, Database, Schema, Warehouse

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Which element of the worksheet context is NOT displayed as part of the SnowSQL prompt?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: Which element of the worksheet context is NOT displayed as part of the SnowSQL prompt?

A: Role

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: In what increment are you billed for compute?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: In what increment are you billed for compute?

A: Per second, with a one-minute minimum

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What is a micro-partition?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What is a micro-partition?

A: A small file that stores part of a table, along with metadata about what it contains, such as MIN, MAX, COUNT.



QUESTION

Q: How are you billed for storage?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How are you billed for storage?

A: Per terabyte, per month

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: In the following scenario, for how much time will you be billed for compute?

- Your auto-resume warehouse is set to auto-suspend after 10 minutes, and is initially suspended
- You run a query using that warehouse, and it takes 7 minutes
- You go to lunch and come back a few hours later
 - No one else has used the warehouse while you were gone
- You run a query that takes 43 seconds
- You go home for the day
 - No one else uses the warehouse



QUESTION

Q: In the following scenario, for how much time will you be billed for compute?

- Your auto-resume warehouse is set to auto-suspend after 10 minutes, and is initially suspended
- You run a query using that warehouse, and it takes 7 minutes – **charge 7 minutes**
- You go to lunch and come back a few hours later – **charge 10 minutes waiting for auto-suspend**
 - No one else has used the warehouse while you were gone
- You run a query that takes 43 seconds
- You go home for the day
 - No one else uses the warehouse

charge 10:43

A: 27 minutes, 43 seconds



QUESTION

Q: What are some of the things the cloud services layer is responsible for?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: What are some of the things the cloud services layer is responsible for?

A:

- Storing metadata
- Access control (network policies)
- User authentication
- Query optimization
- Transaction control
- Upgrades and patches to software
- Storing query result cache
- Account management



QUESTION

Q: How many clusters are in a standard Medium warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How many **clusters** are in a standard Medium warehouse?

A: One (1)

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How many servers are in a standard Medium warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How many **servers** are in a standard Medium warehouse?

A: Four (4)

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How are regions and availability zones related?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How are regions and availability zones related?

A: Availability zones are inside a region; Snowflake automatically replicates data to three (3) different availability zones in your region.



QUESTION

Q: If your query is running slowly, what should you do?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: If your query is running slowly, what should you do?

A:

1. Check the query profile
 - If query pruning is low, try to make filters more efficient; do you need to cluster the table?
 - Look for unintended cross-joins
 - Look for spilling to local or remote disk, and check memory-intensive clauses (GROUP BY, ORDER BY...)
2. Try with a larger warehouse
 - Especially if you are spilling to remote storage
3. Call Support



QUESTION

Q: TRUE or FALSE: Snowflake can automatically scale your warehouse up when needed
(make your warehouse larger)



QUESTION

- Q:** TRUE or FALSE: Snowflake can automatically scale your warehouse up when needed
(make your warehouse larger)
- A:** FALSE. Snowflake can only automatically scale your warehouse out



QUESTION

Q: What are the two auto-scaling methods for a multi-cluster warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: What are the two auto-scaling policies for a multi-cluster warehouse?

A: Standard and Economy



QUESTION

Q: With Economy scaling, how long might queries be queued before another cluster is added to the warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: With Economy scaling, how long might queries be queued before another cluster is added to the warehouse?

A: Six (6) minutes



QUESTION

Q: What is the difference between a temporary table and a transient table?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: What is the difference between a temporary table and a transient table?

A: A temporary table is only available in a single session, and is dropped when the session ends. A transient table is available across sessions, and kept until it is explicitly dropped.



QUESTION

Q: Why would you "scale up" your warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Why would you "scale up" your warehouse?

A: To provide more resources for complex queries, to improve performance



QUESTION

Q: What ROLE should create users and roles?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What ROLE should create users and roles?

A: USERADMIN or optionally SECURITYADMIN



QUESTION

Q: What privileges does SYSADMIN have by default?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What privileges does SYSADMIN have by default?

A: CREATE DATABASE and CREATE WAREHOUSE



QUESTION

Q: How long does the query result cache last?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: How long does the query result cache last?

A: 24 hours, with the timer “reset” each time the cached result is used



QUESTION

Q: Who can use the data cache?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Who can use the data cache?

A: Anyone who uses the same warehouse

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Who can use the query result cache?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Who can use the query result cache?

A: For a SHOW: anyone in the same role

For a SELECT: anyone with select permission on all the tables in the query



QUESTION

Q: For how long does a query remain in the query history tab?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: For how long does a query remain in the query history tab?

A: 14 days

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Name three differences between INFORMATION_SCHEMA and ACCOUNT_USAGE

Roger.Andrei1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 - do-not-copy



QUESTION

Q: Name three differences between INFORMATION_SCHEMA and ACCOUNT_USAGE

A: ACCOUNT_USAGE includes dropped objects; INFORMATION_SCHEMA does not
ACCOUNT_USAGE retains information for one year; INFORMATION_SCHEMA retains
information for up to 6 months (it varies by view)
ACCOUNT_USAGE has up to 3 hours of latency; INFORMATION_SCHEMA has no
latency



QUESTION

Q: What command is used to load or unload data?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What command is used to load or unload data?

A: COPY INTO

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What is the advantage of using an external stage, rather than copying data in directly from the cloud storage?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

- Q:** What is the advantage of using an external stage, rather than copying data in directly from the cloud storage?
- A:** With an external stage, you can define the credentials and keys needed to access and decrypt the data. This allows you to create a stage that can be used without giving the users information on the credentials.



QUESTION

Q: With enterprise edition, what is the longest time you can specify for Time Travel?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: With enterprise edition, what is the longest time you can specify for Time Travel?

A: 90 days

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: Why would you "scale out" your warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Why would you "scale out" your warehouse?

A: To spin up additional clusters, to provide greater concurrency



QUESTION

Q: What is a virtual warehouse?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: What is a virtual warehouse?

A: A collection of compute resources that can be used for queries, loading, etc.



QUESTION

Q: You have Time Travel set to keep your table data for 27 days. 35 days after dropping a table you want to get it back. Can you?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

- Q:** You have Time Travel set to keep your table data for 27 days. 35 days after dropping a table you want to get it back. Can you?
- A:** No. The 7-day Failsafe period would only protect you through day 34.



QUESTION

Q: When you share data with a consumer account, who pays for compute time?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

Q: When you share data with a consumer account, who pays for compute time?

A: The consumer account does.



QUESTION

Q: Can you set a compute quota on a specific user?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Can you set a compute quota on a specific user?

A: No, you can set a compute quota at either the account or warehouse level



QUESTION

Q: Can you set a storage quota at the account level?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020-do-not-copy



QUESTION

Q: Can you set a storage quota at the account level?

A: No, you cannot set storage quotas at any level



QUESTION

Q: Is there any reason you wouldn't want to auto-suspend all your warehouses after one minute?

Roger.Andre1@t-mobile.com 26-Oct-2021 ©Snowflake 2020 do-not-copy



QUESTION

- Q:** Is there any reason you wouldn't want to auto-suspend all your warehouses after one minute?
- A:** Suspending a warehouse clears the data cache. If another user runs a very similar query right after the warehouse is suspended, they will need to re-load all the data to the warehouse, which might take more time than you saved by shutting the warehouse down.





snowflake®

THANK YOU



Roger.Andre1@t-mobile.com 26-Oct-2021 © Snowflake 2020-do-not-copy



© 2021 Snowflake Computing Inc. All Rights Reserved