MockCert

Study guide | Snowflake | COF-C03

SnowPro Core

Notes for every domain of the SnowPro Core exam, organized by the official exam guide's domains and weights.

By MockCert about 83 min read Updated November 13, 2026

CONTENTS

DOMAIN 1 · 31%
Snowflake AI Data Cloud Features and Architecture 2

DOMAIN 2 · 20%
Account Management and Data Governance 20

DOMAIN 3 · 18%
Data Loading, Unloading, and Connectivity 31

DOMAIN 4 · 21%
Performance Optimization, Querying, and Transformation 40

DOMAIN 5 · 10%
Data Collaboration 51

MockCert is an independent certification preparation platform. It is not affiliated with, endorsed by or operated by Snowflake. Exam domains and weights follow the official exam guide. These notes were last updated on November 13, 2026; the notes at mockcert.com are kept up to date as the exam changes.

Domain 1 | 31% of the exam | 26 min read

Snowflake AI Data Cloud Features and Architecture

Overview

This domain is about 31% of the exam, roughly 31 of its 100 questions.

Every point below comes from Snowflake's official documentation; each section lists its sources.

Describe and use the Snowflake architecture

Start here

THINK FIRST

How do the storage, compute and cloud services layers split the work, and which edition do you need for a feature?

Why it matters: Questions describe a workload, cost or feature need and ask which layer handles it or which edition provides it.

Storage holds table data in micro-partitions, organized into databases and schemas. Compute runs on virtual warehouses and Snowflake-managed serverless resources. Cloud services handles metadata work, and each edition adds features on top of the one before.

Key terms

Term	Meaning
Micro-partition	The unit in which Snowflake stores a table's data, clustered by dimensions of the data.
Cloud services layer	The layer whose compute runs metadata work such as DDL, cloning and SHOW commands.
Namespace	A database and schema together; each schema belongs to a single database.
Compute pool	An account-level Snowpark Container Services construct, much like a virtual warehouse.
Time Travel	Keeps a table's historical data for a retention period: **0** or **1** day on Standard, up to **90** on Enterprise.

The storage layer

Snowflake stores table data in micro-partitions. It clusters them by dimensions of the data. Each database belongs to one account and holds schemas, which group objects such as tables and views. There is no hard limit on databases, schemas or objects.

EXAM TRAP

Time Travel and Fail-safe both hold a table's historical data, but not as a full copy for every change. Full copies exist only when tables are dropped or truncated.

Measuring storage

DATABASE_STORAGE_USAGE_HISTORY reports daily storage per database. It covers **365** days, with latency up to **3** hours. AVERAGE_DATABASE_BYTES includes Time Travel bytes, while AVERAGE_FAILSAFE_BYTES shows Fail-safe. For billed storage, use STORAGE_DAILY_HISTORY instead.

The compute layer

Compute cost has three parts. They are user-managed virtual warehouses, serverless features such as Automatic Clustering and Snowpipe, and the cloud services layer. METERING_DAILY_HISTORY shows how many compute credits were actually billed.

Compute pools

A Snowpark Container Services compute pool works much like a warehouse. Its name must be unique in the account. Choosing an instance family is like choosing a warehouse size, but billing follows the instance type actually provisioned on each node. Stop all running services before dropping a pool.

EXAM TRAP

Suspending a compute pool suspends every service on it except job services. Those keep running until they reach a final state, DONE or FAILED, and only then are the nodes released.

Cloud services

Metadata operations use only cloud services compute. These include DDL such as cloning, SHOW commands and Information Schema queries. Listing files during COPY also uses cloud services, so poorly selective COPY commands raise that usage.

EXAM TRAP

Cloud services usage is not always billed. It is charged only when daily use exceeds **10%** of daily warehouse usage.

Snowflake editions

Each edition builds on the one before it. The edition also sets the unit cost of credits and storage. Needing Enterprise features, such as materialized views, raises the price of each credit.

What each edition adds

Edition	What it adds	Examples
Enterprise	Features for large-scale enterprises	Materialized views, search optimization, Time Travel up to **90** days
Business Critical	More protection for extremely sensitive data, such as PHI	Account failover and failback
Virtual Private Snowflake	A completely separate environment	No hardware shared with accounts outside it

Iceberg table storage

Iceberg tables use Snowflake storage or an external volume. With Snowflake storage, Snowflake stores and manages the table files, so you configure no cloud storage access.

Snowflake storage and external volumes for Iceberg tables

Feature	Snowflake storage	External volume
How it is chosen	EXTERNAL_VOLUME = SNOWFLAKE_MANAGED	Required with an external Iceberg catalog
Fail-safe	Yes, for permanent tables	Not provided
Data compaction	Automatic and free when Snowflake writes	An additional charge

Key takeaways

- DDL, cloning, SHOW and Information Schema queries use only cloud services compute.
- Cloud services are billed only above **10%** of daily warehouse usage.
- Materialized views and search optimization need Enterprise Edition or higher.
- Automatic Clustering adds no storage, while materialized views and search optimization do.

Sources: Optimizing storage for performance | Snowflake Documentation (opens in a new tab) · Database, schema, & share DDL | Snowflake Documentation (opens in a new tab) · Exploring compute cost | Snowflake Documentation (opens in a new tab) · Optimizing cloud services for cost | Snowflake Documentation (opens in a new tab) · Snowflake editions | Snowflake Documentation (opens in a new tab) · Snowpark Container Services: Working with compute pools | Snowflake Documentation (opens in a new tab) · Storage costs for Time Travel and Fail-safe | Snowflake Documentation (opens in a new tab) · Databases, Tables & Views | Snowflake Documentation (opens in a new tab)

Use Snowflake Interfaces and tools

Start here

THINK FIRST

When you work in Snowsight, what decides the role your code runs with, who sees your files and what costs credits?

Why it matters: Questions describe a Snowsight task and ask which role, privilege, limit or cost applies to it.

Workspaces is now the Snowsight editor, a file-based place for SQL and Python. Every worksheet or file runs with its own role context, separate from the role in your user menu. Running code, and some changes to results, uses warehouse credits.

Key terms

Term	Meaning
Workspaces	The file-based Snowsight editor for SQL and Python, with folders and Git integration.
My Workspace	Your private default workspace; it cannot be deleted or renamed.
Shared workspace	A workspace for collaboration, where files added with PUT need a commit before others see them.
Worksheet context	The role and warehouse a worksheet runs with, kept for later sessions and shared with its users.
Secondary roles	Extra roles active beside your primary role; they widen what you can see and do, but not what you can create.

Workspaces replaced worksheets

Workspaces is the Snowsight editor now and cannot be turned off. It replaced Legacy Worksheets, reader accounts included. On first use, Snowflake creates a personal database that stores workspaces but holds no tables or views. Notebooks add cells for Python, SQL and Markdown. Workspace SQL commands run from any client, including SnowSQL and the Snowflake CLI.

EXAM TRAP

You cannot disable Workspaces or switch back to Legacy Worksheets, not even in a reader account.

Private and shared workspaces

My Workspace is private; shared workspaces are for teamwork. In SQL, My Workspace is named `USER$.PUBLIC.DEFAULT$`. Granting WRITE on a workspace also grants READ. `ALTER WORKSPACE … COMMIT` has no conflict protection, so it can overwrite files others changed recently.

Private and shared workspaces compared

Feature	My Workspace	Shared workspace
Who can open it	Only you	Other users, for collaboration
Files added with PUT	Visible at once, no commit needed	Hidden from others until committed

EXAM TRAP

REMOVE does not wait for a commit the way PUT does: in a shared workspace the file disappears for everyone right away.

Roles in worksheets and files

Each worksheet or file keeps its own role. Changing your active role in the user menu leaves a worksheet's role as it was. In Workspaces, a file's role selector affects only that file. The selector sets the primary role, so run `USE SECONDARY ROLES` to add secondary roles. Object creation always follows the primary role.

EXAM TRAP

Switching your active role in the user menu does not switch an open worksheet's role, because each worksheet has its own session.

Worksheet context and folders

A worksheet's context is saved and shared with all its users. Context means the role and warehouse the worksheet uses. Folders no longer carry a role, and a new worksheet in a folder gets your session's role. To resume or resize a warehouse from the context selector, you need MODIFY or OWNERSHIP on it.

EXAM TRAP

Not every user can resize or resume a warehouse from the context selector; it takes MODIFY or OWNERSHIP on that warehouse.

Results and compute cost

Running code uses credits, and so do some result changes. Sorting a result column applies to all results and is billed to the warehouse that ran the query. Formatting, such as thousands separators, percentages, decimal precision and dates, costs nothing. To find billed result changes, filter Query History for `snowsight_transform_cte`.

EXAM TRAP

Changing decimal precision or date format in results costs no compute, but sorting a column does.

Numbers to know

Snowsight limits show up as exact numbers. Some query details, such as duration and data scanned, stay available for only **14** days. Snowflake does not limit the size of results you download as CSV or TSV.

Snowsight limits

Feature	Limit	Detail
Worksheet query history	Up to **25** queries	From the last **14** days; use the worksheet's role to view it

Feature	Limit	Detail
File upload to load a table	**50MB** or less per file	Structured or unstructured files
Contextual statistics	Up to **1 million** result rows	For any selection, plus overall statistics
Result rows shown	**10,000** rows	Still applies in U.S. government regions, Virtual Private Snowflake (VPS) and Private Connectivity accounts
Pinned shortcuts	Up to **3** pages	Pinned to the Shortcuts menu

EXAM TRAP

Accounts in U.S. government regions, on Virtual Private Snowflake (VPS) or using Private Connectivity still see query results capped at **10,000** rows.

Browsers and network access

Snowsight needs a supported browser and an allowed network path. It supports the latest **three** major versions of each supported browser. Behind a proxy or firewall, read the `SNOWSIGHT_DEPLOYMENT` entry from `SYSTEM$ALLOWLIST`. For private connectivity, an ACCOUNTADMIN calls `SYSTEM$GET_PRIVATELINK_CONFIG`. A user who dismisses MFA enrollment is asked again every **three** days.

Example: Lists the endpoints to allow, including the SNOWSIGHT_DEPLOYMENT entry for Snowsight.

```
SELECT SYSTEM$ALLOWLIST();
```

Governance and navigation

Snowsight manages policies, roles and security checks. Masking policies secure columns, row access policies filter rows, and the Trust Center evaluates account security. Catalog » Explorer shows only objects your active role holds at least USAGE on. Warehouses now sit under Compute » Warehouses, and an X-Small warehouse generally serves features like Data Preview.

Key takeaways

- Workspaces replaced Legacy Worksheets and cannot be disabled.
- A worksheet keeps its own role; the role in your user menu does not change it.
- Object creation uses only your primary role, even with secondary roles active.
- Sorting results costs compute, while formatting results does not.
- In a shared workspace, PUT needs a commit, but REMOVE publishes at once.

Sources: Workspaces | Snowflake Documentation (opens in a new tab) · Snowsight quick tour | Snowflake Documentation (opens in a new tab) · Work with worksheets in Snowsight | Snowflake Documentation (opens in a new tab) · Querying data using worksheets | Snowflake Documentation (opens in a new tab) · Workspace operations using SQL | Snowflake Documentation (opens in a new tab) · Getting started with Snowsight | Snowflake Documentation (opens in a new tab) · Snowsight navigation menu | Snowflake

Documentation (opens in a new tab) · Explore and manage database objects in Snowsight | Snowflake Documentation (opens in a new tab)

Differentiate Snowflake object hierarchy and types

Start here

THINK FIRST

How do you tell which Snowflake object or hierarchy level handles a task, such as creating accounts, returning a value, or replicating files?

Why it matters: Questions describe a task or a limit and ask which object type, or which level of the hierarchy, applies to it.

An organization sits at the top and holds accounts. Each account keeps its own users, roles, databases and warehouses. Object types then differ by job: a file format helps load and unload data, while a stored procedure runs SQL statements for administrative work.

Key terms

Term	Meaning
Organization account	The single account an organization uses for organization-wide tasks.
GLOBALORGADMIN	The role used in the organization account for all organization-level tasks.
UDF	A user-defined function; one SQL statement can call many of them.
Stored procedure	An object, run with CALL, that performs administrative operations through SQL statements.
File format	A named object used when loading or unloading data.

Organization and accounts

One organization account sits above the regular accounts. Each account still has its own users, roles, databases and warehouses. Usage in every account lands on a single bill. Administrators use the GLOBALORGADMIN role in the organization account; using ORGADMIN in an ORGADMIN-enabled account is being phased out.

EXAM TRAP

You cannot convert an existing ORGADMIN-enabled account into the organization account. An organization has only one organization account.

Creating an account

The organization administrator picks the cloud, region and edition. The region must already be enabled for the organization. The first user you name gets the ACCOUNTADMIN role. By default an organization can

hold **25** On Demand accounts, or **100** with a capacity contract. Allow about **30** seconds for DNS before you connect.

UDFs and stored procedures

A UDF returns a value; a stored procedure performs operations. Choose a stored procedure for administrative work through SQL statements. Choose a UDF when a query needs a value back. UDF handlers can be written in Java, JavaScript, Python, Scala, or SQL and Snowflake Scripting.

How UDFs and stored procedures differ

Feature	UDF	Stored procedure
Calls in one SQL statement	Many UDFs	Only one procedure
SQL it can run	Queries only, no DML or DDL	SQL statements for administrative operations
Return value	Always returns a value	Optional; returns NULL when none is returned
How it runs	Called inside a SQL statement	Runs as its own statement with CALL

EXAM TRAP

A procedure that returns nothing still needs a RETURNS clause in its CREATE PROCEDURE statement.

Stages and replication

External stage files are not copied by replication. Primary and secondary databases point to the same cloud storage URL for an external stage. A directory table is an option you enable on the stage itself. A named internal stage can have its files replicated, but only when that option is enabled on it.

EXAM TRAP

Do not expect files on a user stage, a table stage or an external stage to replicate. A named internal stage replicates its files only with a directory table enabled on it.

File formats in Account Usage

The FILE_FORMATS view lists file format objects. Its latency can reach **120** minutes. It does not recognize the MANAGE GRANTS privilege, so a SHOW command run by a holder of that privilege may show more.

Custom session attributes

Custom session attributes cannot change once set. They last for the whole session, and their names are case-sensitive, so `app_context` and `APP_CONTEXT` differ. Any user can read attributes from their own session without special privileges. `SYS_CONTEXT` returns a VARCHAR, or NULL when the attribute was never set.

EXAM TRAP

An owner's rights stored procedure can call `SYS_CONTEXT` with the `SNOWFLAKE$SESSION` namespace only if the owner role holds READ SESSION on the account.

Changing an account edition

Organization administrators change editions with ALTER ACCOUNT. The command accepts STANDARD, ENTERPRISE or BUSINESS_CRITICAL; VPS needs Snowflake Support. Before a downgrade, stop using features unique to the higher edition, such as masking policies. You cannot downgrade within the same day as a change.

EXAM TRAP

Reader accounts and other managed accounts cannot use ALTER ACCOUNT ... SET EDITION. Snowflake Support changes their edition.

Key takeaways

- An organization has **one** organization account, where GLOBALORGADMIN handles organization-level tasks.
- One SQL statement can call many UDFs but only **one** stored procedure.
- User, table and external stage files are not replicated, and a stage needs a directory table enabled for its files to replicate.
- Custom session attributes are immutable and last for the whole session.

Sources: Organization accounts | Snowflake Documentation (opens in a new tab) · Creating an account | Snowflake Documentation (opens in a new tab) · FILE_FORMATS view | Snowflake Documentation (opens in a new tab) · Choosing whether to write a stored procedure or a user-defined function | Snowflake Documentation (opens in a new tab) · Stage, pipe, and load history replication | Snowflake Documentation (opens in a new tab) · SYS_CONTEXT (SNOWFLAKE$SESSION_ATTRIBUTES namespace) | Snowflake Documentation (opens in a new tab) · SYS_CONTEXT (SNOWFLAKE$SESSION namespace) | Snowflake Documentation (opens in a new tab) · Working with account editions | Snowflake Documentation (opens in a new tab)

Configure virtual warehouses

Start here

THINK FIRST

How do you configure a warehouse for a workload: a bigger size, more clusters, another type, or a different auto-suspend?

Why it matters: Questions describe a workload and ask which warehouse type, size, scaling policy or setting fits it.

Resizing helps slow, complex queries and large loads, while adding clusters helps concurrency. The

warehouse type sets memory per node and, for standard warehouses, the generation. Auto-suspend trades saved credits against a warm cache.

Key terms

Term	Meaning
Snowpark-optimized warehouse	A warehouse type with **16x** memory per node by default, for memory-heavy Snowpark code.
Multi-cluster warehouse	A warehouse that adds clusters to handle more concurrent users and queries.
Scaling policy	The rule for starting clusters: Standard favors less queuing, Economy favors saving credits.
AUTO_SUSPEND	Seconds of inactivity before a running warehouse suspends; null means it never suspends.
Adaptive Warehouse	A warehouse, often converted from an existing one, tuned by a query performance level and a throughput multiplier.

Size up or scale out

Size up for complex queries; scale out for concurrency. Multi-cluster warehouses do little for slow queries or data loading, where resizing helps more. Resizing a running warehouse does not affect queries already running. Resizing a suspended one provisions nothing until it resumes.

EXAM TRAP

A bigger warehouse is rarely needed just to load data. Unless hundreds or thousands of files load at once, Small, Medium or Large is usually enough.

What size costs

Each larger size doubles credit usage. Snowflake bills by the second, but every start costs at least **60** seconds. Consumption is still reported in hourly increments. Add WAIT_FOR_COMPLETION to ALTER WAREHOUSE to confirm that a resize is fully provisioned.

Scaling policies

Standard, the default policy, starts clusters to avoid queuing. Auto-scale mode runs when the minimum and maximum cluster counts differ; equal counts mean Maximized mode. A single-cluster warehouse always has **1** for both counts.

Standard and Economy scaling policies

Feature	Standard	Economy
Favors	Starting clusters over saving credits	Saving credits by keeping clusters fully loaded
Starts a cluster	To prevent or minimize queuing	Only with load for at least **6** minutes

EXAM TRAP

An underused multi-cluster warehouse wastes credits through its minimum cluster count. Lower MIN_CLUSTER_COUNT to cut that waste.

Snowpark-optimized warehouses

Snowpark-optimized warehouses suit memory-heavy or CPU-specific code. ML training in a stored procedure is a typical case. The default gives **16x** memory per node, with resource constraint MEMORY_16X. Other values include MEMORY_1X and MEMORY_64X, each also offered as an x86 variant.

EXAM TRAP

The Gen1 or Gen2 setting applies only to standard warehouses, never to Snowpark-optimized ones.

Gen1 and Gen2 standard warehouses

A standard warehouse can switch between Gen1 and Gen2. The switch works whether the warehouse is running or suspended. Gen2 is not offered in every region, or in sizes X5LARGE and X6LARGE. Set the GENERATION clause with SQL, because Snowsight does not offer it. A new Gen2 warehouse turns on the Query Acceleration Service, which eases the impact of outlier queries. Converting from Gen1 leaves it unchanged.

EXAM TRAP

Converting a running Gen1 warehouse lets existing queries finish on Gen1 while new ones use Gen2. You pay for both sets of compute meanwhile.

Concurrency settings

MAX_CONCURRENCY_LEVEL defaults to 8. Lowering it can speed up large, complex or multi-statement queries, but test the change first. It can also put more queries in the queue. STATEMENT_QUEUED_TIMEOUT_IN_SECONDS cancels queued queries instead of letting them wait a long time.

Auto-suspend

Auto-suspend is on by default. AUTO_SUSPEND is a number of seconds, and null means never suspend. Snowflake suggests a low value, such as **5** or **10** minutes or less. Suspending drops the data cache, so the first queries after a resume can be slower.

Separating teams and workloads

Separate warehouses keep workloads such as analytics and ETL apart. For example, finance and engineering can each use their own Adaptive Warehouse. Adaptive Warehouses need Enterprise Edition or higher. Interactive warehouses are a separate type to consider when dashboards, applications or other user-facing queries need very low latency.

Key takeaways

- Size up for complex queries and big loads; add clusters for concurrency.
- Each larger size doubles credits, billed per second with a **60**-second minimum.
- Economy starts a cluster only for at least **6** minutes of load; Standard favors less queuing.
- Snowpark-optimized warehouses give **16x** memory per node by default.

Sources: Working with warehouses | Snowflake Documentation (opens in a new tab) · Multi-cluster warehouses | Snowflake Documentation (opens in a new tab) · Snowpark-optimized warehouses | Snowflake Documentation (opens in a new tab) · Snowflake generation 2 standard warehouses | Snowflake Documentation (opens in a new tab) · Warehouse considerations | Snowflake Documentation (opens in a new tab) · SHOW WAREHOUSES | Snowflake Documentation (opens in a new tab) · Adaptive Compute | Snowflake Documentation (opens in a new tab) · Overview of warehouses | Snowflake Documentation (opens in a new tab)

Explain Snowflake storage concepts

Start here

THINK FIRST

How do you pick the right table or view type for a workload, and when is a clustering key worth its cost?

Why it matters: Questions describe a data, freshness or sharing need and ask which table type, view type or clustering choice fits it.

Snowflake splits every table into compressed micro-partitions on its own, in load order. Table types differ in lifetime, recovery and where files live, while view types trade speed for security. A clustering key regroups rows into new micro-partitions, at a credit and storage cost.

Key terms

Term	Meaning
Micro-partition	The automatic storage unit of every table, holding **50 MB** to **500 MB** of uncompressed data.
Clustering depth	A measure of clustering quality: the smaller the average depth, the better the table is clustered.
Fail-safe	A data protection period that permanent tables have and transient tables lack; no table type lets you configure it.
Target lag	How stale a dynamic table's data may become, not how often it refreshes.
Secure view	A view whose definition only the owning role sees, and whose internals Query Profile hides even from the owner.

Micro-partitions

Every table is split into micro-partitions automatically. Snowflake partitions rows in the order they are inserted or loaded. Each micro-partition holds **50 MB** to **500 MB** of uncompressed data, yet storage is always compressed. Snowflake will not prune (skip) micro-partitions for a filter built on a subquery, even one returning a constant.

EXAM TRAP

A subquery that returns a constant does not help pruning: Snowflake never prunes micro-partitions on a predicate that contains a subquery.

Clustering keys and depth

Smaller average clustering depth means a better clustered table. Monitor clustering with `SYSTEM$CLUSTERING_DEPTH` or `SYSTEM$CLUSTERING_INFORMATION`. In a multi-column key, order columns from lowest to highest cardinality. For a very high-cardinality column, such as a timestamp, cluster on an expression like a cast to a date.

EXAM TRAP

Putting the highest-cardinality column first is backward: the general rule orders key columns from lowest to highest cardinality.

Automatic Clustering and Optima

Automatic Clustering spends credits but needs no warehouse of yours. It never blocks DML. Reclustering writes new micro-partitions but keeps the old ones, so storage can grow. They stay for Fail-safe and for Time Travel, a table's retention period for past data. A suspended table is not reclustered or charged. Optima Clustering bills on uncompressed data ingested. Changing a key with `ALTER TABLE` does not move a table to Optima, and no migrate command exists.

Clustering Classic and Optima Clustering compared

Feature	Clustering Classic	Optima Clustering
Which tables use it	Tables already clustered before September 1, 2026	Tables newly clustered from September 1, 2026
Clustering key length used	First **5 bytes** of each key column	Up to **1 KB** across all key columns

EXAM TRAP

A resource monitor cannot cap Automatic Clustering credits; it does not control the Snowflake-provided AUTOMATIC_CLUSTERING warehouse.

Keys when tables are copied

A clone copies the clustering key; CTAS does not. After a clone, Automatic Clustering is suspended until you resume it. A new key leaves existing rows alone until Snowflake reclusters. Hybrid tables cannot have a clustering key at all.

What happens to a clustering key

Action	Clustering key	What follows
`CREATE TABLE … AS SELECT`	Not carried over	Define a key after the table exists
`CREATE TABLE … CLONE`	Copied	Automatic Clustering is suspended until resumed
Key changed with `ALTER TABLE`	New key set	Existing rows change only after reclustering
Hybrid table	Not allowed	Rows always ordered by primary key

EXAM TRAP

A cloned table is not reclustered on its own: the key comes along, but Automatic Clustering stays suspended until you resume it.

Permanent, transient and temporary

Lifetime and Fail-safe set the three native table types apart. Permanent tables keep **0** to **90** days of Time Travel on Enterprise Edition and higher. A temporary table is seen only by its session and overrides a same-named table in that schema. Tables in a transient schema or database are transient, and a transient table can never change type.

Native table types compared

Feature	Permanent	Transient	Temporary
Lasts until	Explicitly dropped	Explicitly dropped	The creating session ends
Fail-safe	**7** days	None	Data purged at session end, not recoverable

EXAM TRAP

You cannot shorten or switch off Fail-safe on any table type; to avoid it, create the table as transient from the start.

External and Iceberg tables

External tables are read-only; Iceberg tables choose who stores files. You can query, join and build views on an external table, but not run DML on it. Its queries may be slower than native tables, and a materialized view can help. With Snowflake storage, Snowflake manages Iceberg data and metadata files and gives

permanent Iceberg tables Fail-safe. An external Iceberg catalog requires external volume storage.

EXAM TRAP

An external table takes no INSERT, UPDATE or DELETE: it is read-only, though queries, joins and views on it still work.

Dynamic tables

A dynamic table keeps a query's results fresh within a target lag. Target lag caps staleness; it is not a schedule. A lag of **10 minutes** means data at most **10 minutes** old. The minimum is **60 seconds**, and the definition cannot use stored procedures or external functions. With `TARGET_LAG = DOWNSTREAM`, it refreshes only when a downstream dynamic table triggers it.

EXAM TRAP

A `DOWNSTREAM` dynamic table with no downstream consumers never refreshes on its own, and Snowflake gives no warning.

Materialized and secure views

Materialized views read one table; secure views hide their definition. A materialized view cannot join, not even to itself. Only the owning role sees a secure view's definition, and Query Profile hides its internals even from the owner. Secure views can run slower, so use them to protect data. In shares, `CURRENT_ROLE` and `CURRENT_USER` return NULL; test with `SIMULATED_DATA_SHARING_CONSUMER`.

Example: Turns an existing view into a secure view.

```
ALTER VIEW my_view SET SECURE;
```

- `SET SECURE`: UNSET SECURE turns it back into a regular view

EXAM TRAP

Sharing tables directly is not the safe choice; Snowflake strongly recommends sharing secure views or secure UDFs instead.

Key takeaways

- Micro-partitioning is automatic and follows load order, at **50 MB** to **500 MB** uncompressed each.
- Transient tables skip Fail-safe and can never be converted to another table type.
- Automatic Clustering bills credits without your warehouse, and resource monitors cannot cap it.
- Target lag sets maximum staleness, and **60 seconds** is the lowest it goes.
- Secure views show their definition only to the owning role, hide internals in Query Profile, and can run slower.

Sources: Micro-partitions & Data Clustering | Snowflake Documentation (opens in a new tab) · Working with Temporary and Transient Tables | Snowflake Documentation (opens in a new tab) · Working with Secure Views | Snowflake Documentation (opens in a new tab) · Clustering Keys & Clustered Tables | Snowflake

Documentation (opens in a new tab) · Automatic Clustering | Snowflake Documentation (opens in a new tab) · Storage for Apache Iceberg™ tables | Snowflake Documentation (opens in a new tab) · Set the target lag for a dynamic table | Snowflake Documentation (opens in a new tab) · Introduction to external tables | Snowflake Documentation (opens in a new tab)

Explain AI/ML and application development features

Start here

THINK FIRST

When a workload needs generated text, fuzzy search over documents, or answers from business data, which Cortex feature fits it?

Why it matters: Questions describe an AI need and ask which Cortex function, service option, limit or cost applies to it.

In Cortex, SENTIMENT scores text from **-1** to **1**, and CLASSIFY_TEXT returns an object whose label names a category. COMPLETE is a legacy function; new work starts with AI_COMPLETE. Cortex Search gives fuzzy search over text, and Cortex Analyst works from a semantic model stored in a stage file or a semantic view.

Key terms

Term	Meaning
AI_COMPLETE	The canonical text-generation function for new use cases; COMPLETE is its legacy form.
Cortex Search Service	An object that gives low-latency fuzzy search over text data, with extra storage and compute costs.
TARGET_LAG	The longest time Cortex Search content may trail behind updates to its base tables.
Semantic model	What Cortex Analyst works from, loaded from a file on a stage or from a semantic view.

Generating text with COMPLETE

COMPLETE keeps no state between calls. For a conversation, pass every earlier prompt and response, so tokens and cost grow each round. Only one system prompt is allowed, and it must come first. Calling it needs a role granted `SNOWFLAKE.CORTEX_USER`.

EXAM TRAP

COMPLETE does not remember the previous call. If the history is not in the prompt array, the model sees only the new prompt.

COMPLETE options and errors

TRY_COMPLETE returns NULL where COMPLETE raises an error. Start new work with AI_COMPLETE instead. In COMPLETE, temperature defaults to **0**, and max_tokens defaults to **4096** with a maximum of **8192** . The guardrails option, which uses Cortex Guard, defaults to FALSE.

COMPLETE and TRY_COMPLETE side by side

Feature	COMPLETE	TRY_COMPLETE
When the operation fails	Raises an error	Returns NULL
Status	Legacy; deprecated by the end of **2026**	Legacy; deprecated by the end of **2026**

Scoring and classifying text

SENTIMENT returns a score from -1 to 1. Near **1** is positive, near **-1** negative, and near **0** neutral. The score shows polarity and certainty, not intensity. CLASSIFY_TEXT takes **2** to **100** unique categories and returns an object whose label field names the category.

EXAM TRAP

CLASSIFY_TEXT input is case sensitive, so the same text in different capitals can get different results. Category descriptions and examples also count as input tokens.

Parsing documents

PARSE_DOCUMENT defaults to OCR mode. The output is a JSON-formatted string, so convert it with PARSE_JSON before using it as an OBJECT. The page_split option defaults to FALSE and works only for PDF, .pptx and .docx files.

The two PARSE_DOCUMENT modes

Feature	OCR	LAYOUT
What it extracts	Text only	Text and layout, including tables
Default mode	Yes	No

Building a Cortex Search Service

The service's warehouse builds and refreshes the search index. TARGET_LAG caps how far the content may trail the base tables. The default refresh mode is INCREMENTAL, which needs change tracking on all underlying objects. The default embedding model is `snowflake-arctic-embed-m-v1.5`.

EXAM TRAP

A search service keeps adding storage and compute costs. AUTO_SUSPEND is disabled by default, and its minimum is **1800** seconds.

Querying Cortex Search

Filters work only on the ATTRIBUTES columns named at creation. Operators are `@eq`, `@contains`, `@gte`, `@lte` and `@primarykey`, combined with `@and`, `@or` and `@not`. Without a limit, a query returns at most **10** results. SEARCH_PREVIEW truncates results over **300** kB; the REST surface allows **10** MB.

> **EXAM TRAP**
>
> Do not serve a low-latency end-user app with SEARCH_PREVIEW. It is meant for testing and is slower than the REST or Python APIs.

Batch search

Batch search runs on its own serverless compute. That compute scales apart from interactive search traffic. For fewer than **2,000** queries, the interactive API is usually faster.

Cortex Analyst

Cortex Analyst is billed by the number of messages processed. CORTEX_ANALYST_USAGE_HISTORY shows its credits in one-hour increments for the last **365** days. For a dimension with about **1** to **10** values, list sample values. Above **10** values, or for changing values, use a Cortex Search Service.

Key takeaways

- Use AI_COMPLETE for new work; COMPLETE and TRY_COMPLETE are legacy functions.
- COMPLETE keeps no state, so conversations resend the full history and cost more each round.
- Cortex Search filters apply only to ATTRIBUTES columns, and queries return **10** results by default.
- Cortex Analyst credits depend on the number of messages processed.

Sources: COMPLETE (SNOWFLAKE.CORTEX) | Snowflake Documentation (opens in a new tab) · SENTIMENT (SNOWFLAKE.CORTEX) | Snowflake Documentation (opens in a new tab) · CLASSIFY_TEXT (SNOWFLAKE.CORTEX) | Snowflake Documentation (opens in a new tab) · Improve literal search to enhance Cortex Analyst responses | Snowflake Documentation (opens in a new tab) · CORTEX_ANALYST_REQUESTS_V view | Snowflake Documentation (opens in a new tab) · CREATE CORTEX SEARCH SERVICE | Snowflake Documentation (opens in a new tab) · TRY_COMPLETE (SNOWFLAKE.CORTEX) | Snowflake Documentation (opens in a new tab) · PARSE_DOCUMENT (SNOWFLAKE.CORTEX) | Snowflake Documentation (opens in a new tab)

Domain 2 | 20% of the exam | 16 min read

Account Management and Data Governance

Overview

This domain is about 20% of the exam, roughly 20 of its 100 questions.

Every point below comes from Snowflake's official documentation; each section lists its sources.

Explain Snowflake security model and principles

Start here

THINK FIRST

How do roles, ownership and policies combine to decide whether a Snowflake user can sign in and run a given statement?

Why it matters: Questions describe an access or login problem and ask which role, grant or policy explains it or fixes it.

Snowflake grants privileges to roles, and every object has one owning role that can share access. Roles inherit privileges upward through a hierarchy. Network policies are checked before authentication policies, and a blocked IP stops evaluation.

Key terms

Term	Meaning
Securable object	An object owned by a single role, which by default is the role used to create it.
Role hierarchy	Roles granted to other roles, so a parent role inherits every privilege of the roles below it.
Managed access schema	A schema where only its owner or a role with MANAGE GRANTS can grant privileges on its objects.
Secondary roles	Roles activated in a session beside the primary role; CREATE statements still rely on the primary role.
Network policy	A rule whose blocked list stops requests by IP address once it is activated for an account, user or security integration.

Two access models together

Access comes from roles (RBAC) and from object owners (DAC). Privileges go to roles, and roles go to users. Each object also has one owning role that can grant access to it. A privilege granted to a role flows up to every parent role in the hierarchy.

Role-based and discretionary access control compared

Feature	Role-based (RBAC)	Discretionary (DAC)
Core idea	Privileges are given to roles, which are given to users	Every object has an owner that can share access to it
What decides access	Privileges held by the role or inherited from roles below it	The owning role, by default the role that created the object

EXAM TRAP

Owning a role is not the same as being above it in the hierarchy. The owner role does not inherit the owned role's privileges; only a role hierarchy passes privileges on.

System-defined roles

ACCOUNTADMIN sits on top and contains SYSADMIN and SECURITYADMIN. USERADMIN holds CREATE USER and CREATE ROLE. These built-in roles can never be dropped, and their built-in privileges cannot be revoked. Keep ACCOUNTADMIN to a few users, require MFA for them, and give them a lower default role such as SYSADMIN.

EXAM TRAP

MANAGE GRANTS is not held by every admin role. By default only ACCOUNTADMIN and SECURITYADMIN have it, and it lets the holder see every object.

Account, database and custom roles

Database roles work only after they are granted to an account role. A session can activate account roles but never a database role directly. Build custom roles as a hierarchy whose top role is granted to SYSADMIN. Creating any role needs USERADMIN or higher, or CREATE ROLE on the account.

EXAM TRAP

Seeing a database role is not the same as using it. Any privilege on the database lets a role list its database roles, but they still must be granted to an account role. They also do not appear in the roles graph.

Granting access to objects

Object privileges also need access to the database and schema around them. A role needs USAGE or another privilege on the container database and schema before it can work with an object inside. In a managed access schema, object owners lose grant rights. A holder of MANAGE GRANTS can even grant

privileges straight to users.

Example: Gives a role SELECT on every table that already exists in one schema.

```
GRANT SELECT ON ALL TABLES IN SCHEMA my_db.my_schema
  TO ROLE my_role;
```

- `ON ALL TABLES IN SCHEMA`: covers existing tables only, not tables created later

EXAM TRAP

A grant on all tables in a schema does not cover tables created afterward. Those need a separate grant on future tables.

Diagnosing missing privileges

EXPLAIN_PRIVILEGES names exactly which privileges a failed statement lacks. To run it on your own query, you need at least **one** privilege on the objects involved. Set `missing_only` to true to list only what is still needed. With `for_role`, it checks that role together with its granted roles.

EXAM TRAP

Without at least one privilege on the objects, you cannot call this function on your query. Grant RESOLVE ALL ON ACCOUNT to someone who must diagnose access problems without them.

Primary and secondary roles

CREATE statements are always authorized by the primary role. At login, the user's default role and default secondary roles become active automatically. `USE SECONDARY ROLES` with ALL or NONE then changes how secondary roles are handled. Even with secondary roles active, creating an object relies on the primary role.

The two USE SECONDARY ROLES settings compared

Feature	ALL	NONE
Secondary roles in use	Worked out again as each statement runs	Turned off for the session
Who authorizes CREATE statements	The primary role, even with secondary roles active	The primary role, which authorizes all SQL actions

EXAM TRAP

ALL does not lock in the roles you hold when you run it. Role grants are checked per statement, so a newly granted role becomes active without another command.

Network and authentication policies

Network policies are checked first, and they do nothing until activated. If the network policy blocks a request's IP address, evaluation stops there. Authentication policies then set allowed methods, such as SAML, OIDC, OAuth, key pairs and passwords. MFA second factors include passkeys, authenticator apps and

Duo.

Network and authentication policies compared

Feature	Network policy	Authentication policy
Order of checks	Evaluated first; a blocked IP ends evaluation	Checked only after the network policy lets the request through
Most specific level wins	Security integration overrides both account and user	User-level policy overrides the account-level policy

EXAM TRAP

Blocking client types in an authentication policy is not a firm security boundary. CLIENT_TYPES is best effort, so pair it with other controls.

Logging to an event table

Logged messages from handler code land in an event table. Telemetry collection must be turned on before any messages are collected. The log level controls which severities are kept. Snowflake Scripting logs with SYSTEM$LOG. Request and response translators in external functions cannot log at all.

EXAM TRAP

A log call inside a UDF's row handler is not written once per query. It runs for every row, so a large table can flood the event table.

Key takeaways

- Privileges go to roles, roles go to users, and parent roles inherit from child roles.
- Database roles must be granted to an account role before a session can use them.
- CREATE statements are always authorized by the primary role, even with secondary roles active.
- A network policy blocking an IP stops evaluation before any authentication policy is checked.
- Use EXPLAIN_PRIVILEGES to find exactly which privileges a failed statement is missing.

Sources: Overview of Access Control | Snowflake Documentation (opens in a new tab) · Configuring access control | Snowflake Documentation (opens in a new tab) · Authentication policies | Snowflake Documentation (opens in a new tab) · USE SECONDARY ROLES | Snowflake Documentation (opens in a new tab) · Controlling network traffic with network policies | Snowflake Documentation (opens in a new tab) · Access control best practices | Snowflake Documentation (opens in a new tab) · SHOW DATABASE ROLES | Snowflake Documentation (opens in a new tab) · Troubleshooting access control issues | Snowflake Documentation (opens in a new tab)

Define data governance features and how they are used

Start here

THINK FIRST

How do masking policies, tags, privacy policies and encryption work together to protect sensitive data in Snowflake?

Why it matters: Questions describe a governance need and ask which feature, privilege, limit or key setting meets it.

Snowflake protects data in layers. Masking and aggregation policies control what a query returns, while tags label objects so policies can follow them. Underneath, all customer data is encrypted by default with a managed key hierarchy.

Key terms

Term	Meaning
Masking policy	A schema-level rule that masks column values at query time without changing stored data.
Tag	A schema-level object that stores a string value on an object as a key-value pair.
External Tokenization	Swaps sensitive values for tokens before loading and detokenizes them at query time.
Composite master key	The Tri-Secret Secure key built from a Snowflake key and a customer-managed key.
Entity key	Tells an aggregation policy to count unique entities instead of rows.

Column-level security

Column-level security is Dynamic Data Masking plus External Tokenization. Both rely on masking policies applied at query time. Dynamic Data Masking never rewrites the table, so there is no static masking. External Tokenization stores tokens instead, and an external function restores the values at query runtime.

Dynamic Data Masking and External Tokenization compared

Feature	Dynamic Data Masking	External Tokenization
Stored data	Plain text, left unchanged in the table	Tokens, written before the data is loaded
At query time	A masking policy masks the value	A masking policy with an external function detokenizes it
Collisions	Possible when the policy uses a hash function	None, since each input maps to exactly one token

EXAM TRAP

Dynamic Data Masking never changes what is stored. The table keeps its plain-text values; only query results are masked.

Masking policy rules

A column takes only one masking policy at a time. The policy's input and output data types must match. Editing the policy body updates every column that uses it, with no reapplying. To drop a policy, first unset it from the column with an `ALTER VIEW … MODIFY COLUMN` or `ALTER TABLE … MODIFY COLUMN` statement.

EXAM TRAP

Dropping a masking policy that is still set on a column fails. Unset it on the column first, then run the drop again.

Who can see masked data

Object owners cannot unset a masking policy or see the data it hides. The APPLY privilege on a masking policy lets a role set and unset it. Dynamic Data Masking itself can keep even ACCOUNTADMIN and SECURITYADMIN from viewing protected data needlessly. Use `IS_ROLE_IN_SESSION` when conditions must respect role hierarchy. Inside a policy body, `CURRENT_DATABASE` returns the protected table's database, not the session's.

EXAM TRAP

Owning a table does not let a role read or remove its masking. The OWNERSHIP privilege alone cannot unset a masking policy.

Object tagging basics

A tag is a schema-level object whose value is always a string. One object can carry at most **50** tags. Tags flow down the hierarchy, so a table tag reaches every column. `ALLOWED_VALUES` caps a tag at **5,000** values of up to **256** characters. Removing an allowed value leaves existing assignments as they are. `UNDROP TAG` restores a dropped tag and its assignments within **24** hours.

EXAM TRAP

Automatic propagation of user-defined tags is not in every edition. It requires Enterprise Edition or higher.

Tag-based masking

A policy on a tag protects every tagged column of a matching data type. System tags cannot take masking policies. A table protected this way cannot back a materialized view. When a provider shares a database or schema, its tag-based policies still apply in the consumer account.

Direct and tag-based masking policies compared

Feature	Direct assignment	Tag-based
Set on	A single column	A tag, then every tagged column whose type matches
Limit	One masking policy per column	One masking policy per data type per tag
When both protect a column	Wins over the tag-based policy	Is overridden by the direct policy

EXAM TRAP

When a column has both a direct and a tag-based policy, the tag does not win. The directly assigned policy takes precedence.

Privacy and aggregation policies

Masking policies run before aggregation policies. An aggregation policy sets a minimum group size for query results. With an entity key, that minimum counts unique entities rather than rows. Each entity key maps to at most **1** policy, and only **1** policy may serve row-level privacy.

EXAM TRAP

Masking is not applied to results after they are aggregated. Masking policies run ahead of aggregation policies.

Encryption and keys

All customer data is AES-encrypted by default under a key hierarchy. The hierarchy has **4** levels: root, account master, table master and file keys. Snowflake rotates managed keys once they are over **30** days old. Tri-Secret Secure adds a customer-managed key, forming a composite master key. You register it with `SYSTEM$REGISTER_CMK_INFO`. Revoke it, and Snowflake can no longer decrypt the data.

EXAM TRAP

Files bound for an internal stage are not sent in plain text. The client encrypts them before upload, and they are encrypted again in the stage.

Monitoring and failover

Trust Center's Security Essentials scanners run monthly by default. By default they email verified Admin users only about critical findings. To find every object a masking policy is set on, query the `POLICY_REFERENCES` view. After a failover, failover group objects become writable on the promoted account and read-only on the former primary.

EXAM TRAP

The `QUERY_HISTORY` view does not list masking policy names. Use `POLICY_REFERENCES` to find the objects a policy protects.

Key takeaways

- A column takes one masking policy, and a direct policy beats a tag-based one.
- Object owners cannot unset masking or view masked data; the APPLY privilege controls set and unset.
- An object holds at most **50** tags, and a dropped tag can be undropped within **24** hours.
- Masking policies are enforced before aggregation policies, never after them.
- Revoking the Tri-Secret Secure customer key stops Snowflake from decrypting the data.

Sources: Understanding Column-level Security | Snowflake Documentation (opens in a new tab) · Introduction to object tagging | Snowflake Documentation (opens in a new tab) · Implementing entity-level privacy with aggregation policies | Snowflake Documentation (opens in a new tab) · Understanding Encryption Key Management in Snowflake | Snowflake Documentation (opens in a new tab) · Advanced Column-level Security topics | Snowflake Documentation (opens in a new tab) · Understanding Dynamic Data Masking | Snowflake Documentation (opens in a new tab) · Work with object tags | Snowflake Documentation (opens in a new tab) · Tag-based masking policies | Snowflake Documentation (opens in a new tab)

Explain monitoring and cost management

Start here

THINK FIRST

How would you cap warehouse spending in an account and then work out where its credits actually went?

Why it matters: Questions describe a spending or monitoring need and ask which resource monitor setting, budget or usage view fits it.

Resource monitors set a credit quota on warehouses and act when thresholds are reached. Budgets cover the serverless and AI spending that monitors cannot track. ACCOUNT_USAGE views then keep **1** year of history to explain where credits went.

Key terms

Term	Meaning
Resource monitor	An object that sets a credit quota for warehouses and notifies or suspends them at thresholds.
Credit quota	The credits a monitor allows, counting warehouse use plus the cloud services that support it.
Budget	A spending limit that can also track serverless features and AI services, not only warehouses.
Query load	Query execution time in an interval divided by the interval's length, both in seconds.
ACCOUNT_USAGE	A schema in the shared SNOWFLAKE database that holds usage views such as WAREHOUSE_METERING_HISTORY.

Account and warehouse monitors

An account has one account monitor; each warehouse has at most one monitor. The account monitor watches credit use of every warehouse in the account. A warehouse monitor can watch several warehouses at once. Only ACCOUNTADMIN can create a monitor, but other roles can be granted rights to view and modify monitors.

EXAM TRAP

A warehouse cannot sit under two warehouse monitors for extra protection; it belongs to only **one** resource monitor at a time.

Monitor actions

Suspend lets running queries finish; Suspend Immediate cancels them. A monitor holds **one** Suspend action, **one** Suspend Immediate action and up to **five** Notify actions. Choose Suspend Immediate only when cancelling running statements is acceptable.

The two suspend actions compared

Feature	Suspend	Suspend Immediate
Allowed per monitor	**1**	**1**
Running queries	Allowed to complete	Cancelled at once
Warehouses affected	All assigned warehouses	All assigned standard warehouses

EXAM TRAP

Reaching the Suspend threshold does not kill queries already running; only Suspend Immediate cancels them.

Quota and schedule

The quota counts warehouse credits plus the cloud services behind them. By default, monitoring starts at once and used credits go back to **0** each calendar month. Whatever start time you set, the reset happens at **12:00 AM UTC**.

EXAM TRAP

A monitor set to start at **9:00 AM** local time still resets at **12:00 AM UTC**, not at its own start time.

What monitors cannot see

Resource monitors track warehouses only; use a budget for serverless spend. Snowpipe, automatic reclustering and materialized views fall outside even an account monitor. A budget can track these features and AI services. On warehouses, a budget follows two service types: `WAREHOUSE_METERING` and `QUERY_ACCELERATION`.

EXAM TRAP

An account-level monitor watches every warehouse, yet it does not cap Snowpipe or reclustering credits.

Budgets on shared warehouses

A shared warehouse counts toward a budget only when selected users run queries. Idle time, very short queries, cloud services overhead and automated work such as tasks are not attributed to users. When a warehouse is both direct and shared, the direct measurement wins. Standard budgets can lag up to **11** hours here, and low-latency budgets up to **6**.

Calculating warehouse credits

`CREDITS_USED` **adds compute and cloud services credits, before any adjustment.** It can therefore exceed the billed amount, which `METERING_DAILY_HISTORY` shows. `CREDITS_ATTRIBUTED_COMPUTE_QUERIES` leaves out idle time, so subtract it from `CREDITS_USED_COMPUTE` to estimate idle cost. After a resume, metering starts once all compute resources are provisioned.

EXAM TRAP

`CREDITS_USED` is not the billed figure, because it ignores the cloud services adjustment. Use `METERING_DAILY_HISTORY` for billed credits.

Reading warehouse load

Query load is query execution time divided by the interval's length. Low load with slow queries points to a larger warehouse size. High load or queuing points to a separate warehouse or more clusters. A warehouse running with load below **1** for long periods is wasting credits. Viewing the load chart needs the MONITOR privilege.

ACCOUNT_USAGE versus Information Schema

ACCOUNT_USAGE keeps 1 year of history but lags behind. Information Schema has no lag but keeps far less history. `WAREHOUSE_METERING_HISTORY` returns hourly credits for the last **365** days. `WAREHOUSE_LOAD_HISTORY` reports load in **5**-minute intervals.

ACCOUNT_USAGE and Information Schema compared

Feature	ACCOUNT_USAGE	Information Schema
Latency	Between **45** minutes and **3** hours, per view	None
History kept	**1** year	Between **7** days and **6** months, per view

EXAM TRAP

Information Schema cannot answer a question about last year's spending; it keeps at most **6** months of history.

Key takeaways

- Suspend lets running queries finish, while Suspend Immediate cancels them.
- Monitors reset at **12:00 AM UTC**, whatever start time you choose.
- Use a budget, not a resource monitor, to track serverless features and AI services.
- Check billed credits in `METERING_DAILY_HISTORY`, since `CREDITS_USED` ignores the cloud services adjustment.
- ACCOUNT_USAGE keeps **1** year of history, with up to **3** hours of latency.

Sources: Working with resource monitors | Snowflake Documentation (opens in a new tab) · Account Usage | Snowflake Documentation (opens in a new tab) · RESOURCE_MONITORS view | Snowflake Documentation (opens in a new tab) · Using budgets for warehouses (shared resources) | Snowflake Documentation (opens in a new tab) · Monitoring warehouse load | Snowflake Documentation (opens in a new tab) · Account Overview | Snowflake Documentation (opens in a new tab) · WAREHOUSE_METERING_HISTORY view | Snowflake Documentation (opens in a new tab) · WAREHOUSE_EVENTS_HISTORY view | Snowflake Documentation (opens in a new tab)

Domain 3 | 18% of the exam | 13 min read

Data Loading, Unloading, and Connectivity

Overview

This domain is about 18% of the exam, roughly 18 of its 100 questions.

Every point below comes from Snowflake's official documentation; each section lists its sources.

Perform data loading and unloading

Start here

THINK FIRST

When you load or unload files, how do you choose between an internal stage, an external stage and an external table?

Why it matters: Questions describe a loading, unloading or stage access need and ask which command, option or limit applies to it.

Unloading writes table data out to flat files with `COPY INTO` a location. Internal stages live in Snowflake-managed storage, while external stages point to S3, Azure or GCS. External tables read files from external stages only, never from internal stages.

Key terms

Term	Meaning
Unloading with COPY INTO	`COPY INTO` with a location as its target writes table data out to flat files.
Internal stage	Snowflake-managed storage: on AWS one S3 bucket per deployment, with a prefix for each account.
External table	A table over files in an S3, Azure or GCS stage; internal stages are not supported.
AUTO_REFRESH	An external table property that refreshes metadata through cloud storage event notifications.
Snowpipe Streaming channel	The channel through which the Snowpipe Streaming SDK sends unflushed data to the server for ingestion.

Unloading defaults

Unloaded files use gzip by default and always use `UTF-8`**.** You can unload to delimited files such as CSV or TSV, to JSON, or to Parquet; the default delimiter is a comma. Compression can be turned off or switched to another supported method. No other character set is supported.

Unloading to local files

Local unloads go through an internal location first. Files are written to a Snowflake internal location, and GET downloads them. Internal locations encrypt the files with **128**-bit keys, and the downloaded copies are unencrypted. Files unloaded to cloud storage can be encrypted if you give Snowflake a security key.

EXAM TRAP

Files unloaded to an internal location are never left unencrypted there. Snowflake encrypts them automatically; only the downloaded copies are unencrypted.

Where internal stages live

Internal stages sit in Snowflake-managed cloud storage. On AWS, one S3 bucket serves each internal stage deployment, and a prefix keeps each account's data apart. On Azure, each account gets a dedicated storage account as its internal stage.

Blocking public stage access

A system function, not a network policy, blocks public access. `SYSTEM$BLOCK_INTERNAL_STAGES_PUBLIC_ACCESS` blocks every public IP address; it cannot allow some and block others. `SYSTEM$UNBLOCK_INTERNAL_STAGES_PUBLIC_ACCESS` restores access. `SYSTEM$INTERNAL_STAGES_PUBLIC_ACCESS_STATUS` reports whether public IPs can reach the stage.

EXAM TRAP

The `WITH_EXCEPTION` variant, which allowlists specific IPs or CIDR blocks, works only on Azure, not on AWS or Google Cloud.

Private connectivity to stages

AWS PrivateLink to internal stages needs an account parameter. An account administrator sets `ENABLE_INTERNAL_STAGES_PRIVATELINK` to true. The VPC interface endpoint must be in the same AWS region as the account. On Azure, `SYSTEM$AUTHORIZE_STAGE_PRIVATELINK_ACCESS` authorizes a private endpoint.

External table limits

External tables read files only from external stages. They read any format that `COPY INTO` can load except XML, and the default format type is CSV. Clustering keys, cloning and Time Travel are not supported. File format options on the stage are not inherited, so set them in the table definition.

EXAM TRAP

Do not rely on NOT NULL on an external table. Snowflake enforces no integrity constraints on external tables.

Refreshing external tables

Metadata refreshes by event notification or by a manual REFRESH. The manual form is `ALTER EXTERNAL TABLE … REFRESH`. Automatic refresh is billed as Snowpipe charges. For a location with about **1** million files or more, set `REFRESH_ON_CREATE = FALSE`.

EXAM TRAP

GRANT OWNERSHIP on an external table or its parent database quietly sets AUTO_REFRESH to FALSE. AUTO_REFRESH = TRUE is also unsupported with PARTITION_TYPE = USER_SPECIFIED.

Handling errors

An external table query skips a file it cannot scan. Scanning continues with the next file. The Snowpipe Streaming SDK sends data to the server through a channel for ingestion. The SDK retries **5XX**, **429** and **408** errors itself. A channel whose status is not SUCCESS must be closed and reopened explicitly. After a **401** or **403** error, fix the permission issue before reopening.

Key takeaways

- Unloaded files are gzip-compressed by default and always encoded in `UTF-8`.
- External tables read only external stages and support neither XML, cloning nor Time Travel.
- `SYSTEM$BLOCK_INTERNAL_STAGES_PUBLIC_ACCESS` blocks all public IPs; a network policy is not used for this.
- GRANT OWNERSHIP on an external table turns its AUTO_REFRESH off.

Sources: Summary of Data Unloading Features | Snowflake Documentation (opens in a new tab) · AWS VPC interface endpoints for internal stages | Snowflake Documentation (opens in a new tab) · Azure private endpoints for internal stages | Snowflake Documentation (opens in a new tab) · CREATE EXTERNAL TABLE | Snowflake Documentation (opens in a new tab) · Introduction to external tables | Snowflake Documentation (opens in a new tab) · Error handling in Snowpipe Streaming high-performance architecture | Snowflake Documentation (opens in a new tab) · Refresh external tables automatically | Snowflake Documentation (opens in a new tab)

Perform automated data ingestion

Start here

THINK FIRST

For a pipeline that must stay fresh on its own, when do you use dynamic tables, streams with tasks, or Snowpipe Streaming?

Why it matters: Questions describe an ingestion or pipeline need and ask which feature, setting, privilege or limit fits it.

A dynamic table materializes a SELECT query and keeps it fresh to a target lag. Streams and tasks give finer control, such as one-second schedules and multi-table writes. Snowpipe Streaming appends rows to tables through channels.

Key terms

Term	Meaning
Dynamic table	A table that materializes a SELECT query and keeps it up to date to a target lag.
Target lag	A staleness goal for a dynamic table, not a refresh interval; the minimum is **60** seconds.
Stream	Tracks changes on a base table; a task can check it with SYSTEM$STREAM_HAS_DATA.
Named Channel	A Snowpipe Streaming channel that gives ordered, exactly-once ingestion.

Dynamic tables or streams and tasks

Pick tasks for sub-minute timing or multi-table writes. Dynamic table definitions cannot use stored procedures or external functions. In a task, put SYSTEM$STREAM_HAS_DATA in the WHEN clause so the task runs only when the stream has unconsumed changes.

Dynamic tables compared with streams and tasks

Feature	Dynamic tables	Streams and tasks
Freshness floor	**1** minute target lag	One-second task schedules
Outputs	Exactly one per dynamic table	One task can write many tables in one transaction
Change tracking	Built in on the base table	A stream you create on the base table

Target lag

Target lag is a staleness goal, not a refresh interval. `TARGET_LAG = '10 minutes'` means data at most **10** minutes old. Actual lag can exceed it when refreshes run long. A downstream table's lag must be at least that of the table it reads from.

EXAM TRAP

A dynamic table set to TARGET_LAG = DOWNSTREAM with no downstream consumers never refreshes on its own, and Snowflake gives no warning.

Refresh modes

REFRESH_MODE = AUTO picks a mode once, at creation. It does not re-evaluate later. INTERSECT, EXCEPT and PERCENTILE_CONT need a full refresh. ADAPTIVE refreshes incrementally but reinitializes when a rebuild would be far cheaper. To switch modes, use CREATE OR ALTER or CREATE OR REPLACE, because ALTER DYNAMIC TABLE cannot.

EXAM TRAP

Snowflake never switches a dynamic table's refresh mode on its own, however much of the data changes. To change the mode, you recreate the table with CREATE OR ALTER or CREATE OR REPLACE.

Running refreshes

Refresh results apply atomically, so readers never see partial data. If a refresh is still running, the next one waits instead of overlapping. ALTER DYNAMIC TABLE … REFRESH runs one now, and SCHEDULER = DISABLE leaves refreshes to an external orchestrator. Changing the warehouse or target lag does not reinitialize the table.

Failures and suspension

After 5 scheduled refresh failures in a row, a dynamic table suspends. Manual refresh errors do not count, and failures raise no alert by default. Suspending a table also suspends the dynamic tables downstream. Fixing the error is not enough: run ALTER DYNAMIC TABLE … RESUME.

EXAM TRAP

If an incremental dynamic table stays suspended past its base tables' Time Travel retention, change tracking data expires. The only fix is then to drop and recreate it.

Streams on dynamic tables

Create the stream with ON DYNAMIC TABLE, not ON TABLE. ON TABLE returns an error here. APPEND_ONLY streams are not supported. Dynamic tables that use full refresh support no streams, because each refresh replaces the whole output.

Dynamic table privileges

Creating a dynamic table needs CREATE DYNAMIC TABLE on the schema. You also need USAGE on a warehouse and on the schema. Suspend, resume and alter need OWNERSHIP or OPERATE, while drop and undrop need OWNERSHIP.

Snowpipe Streaming high performance

The high-performance architecture requires a PIPE object. The SDK authenticates with key pairs and always uses UTC. InsertRows is now AppendRows, and ON_ERROR supports only CONTINUE. ARRAY and VARIANT columns need native objects, not strings.

High-performance Snowpipe Streaming limits

Limit	Value	Scope
Table throughput	**20** GB/s uncompressed	Aggregate for one table
Named Channel throughput	**20** MB/s uncompressed	Default for each channel
REST payload	**4** MB	Hard cap per request
Active Named Channels	**2,000**	Default for one pipe

Key takeaways

- Dynamic tables suit pipelines that can lag a minute or more; tasks handle sub-minute timing.
- Target lag is a staleness goal, and actual lag can exceed it.
- After **5** scheduled failures, a dynamic table suspends and must be resumed explicitly.
- High-performance Snowpipe Streaming needs a PIPE object and allows only ON_ERROR = CONTINUE.

Sources: Dynamic tables | Snowflake Documentation (opens in a new tab) · Migrate from streams and tasks to dynamic tables | Snowflake Documentation (opens in a new tab) · Tutorial: Get started with Named Channels using the SDK | Snowflake Documentation (opens in a new tab) · Comparison between Named Channel and classic SDKs | Snowflake Documentation (opens in a new tab) · Set the target lag for a dynamic table | Snowflake Documentation (opens in a new tab) · Use streams on dynamic tables | Snowflake Documentation (opens in a new tab) · Create a dynamic table | Snowflake Documentation (opens in a new tab) · Dynamic table refresh modes | Snowflake Documentation (opens in a new tab)

Identify the different Snowflake Connectors and integrations

Start here

THINK FIRST

When Snowflake must reach cloud storage, an external API or a streaming system, which integration or connector do you set up?

Why it matters: Questions describe a connection need and ask which integration type, privilege, parameter or connector limit applies.

Integrations are named account objects that hold access details, so you avoid passing explicit cloud credentials such as secret keys. Storage integrations serve external stages, while API, notification, catalog and external access integrations serve other services. Openflow connectors bring in data from systems such as Kafka, Kinesis and MySQL.

Key terms

Term	Meaning
Storage integration	An object holding a generated IAM entity for S3, GCS or Azure, plus allowed or blocked locations.
STORAGE_ALLOWED_LOCATIONS	Limits which storage locations the stages using an integration can reference.

Term	Meaning
API integration	Holds details for services reached over HTTPS APIs, such as the cloud platform and credentials.
Openflow connector	A connector that moves data from a system such as Kafka, Kinesis or MySQL into Snowflake.

Integration types

SHOW INTEGRATIONS can list just one integration type. Type filters include API, CATALOG, NOTIFICATION and STORAGE. Only ACCOUNTADMIN has CREATE INTEGRATION by default, but it can be granted to other roles.

What each integration type connects

Integration type	Connects Snowflake to	Holds
Storage	S3, Google Cloud Storage or Azure	An IAM entity and allowed or blocked locations
API	Services reached over HTTPS APIs	Cloud platform, service type and credentials
Notification	Message queues, email services and webhooks	An interface to third-party messaging
Catalog	Apache Iceberg tables	How the Iceberg table metadata is organized
External access	Specific external network locations	Network rules and credentials

Storage integration basics

A storage integration removes explicit cloud credentials from stages. One integration can serve many external stages, and each stage URL must fit STORAGE_ALLOWED_LOCATIONS. With the * wildcard, STORAGE_BLOCKED_LOCATIONS carves out exceptions. On AWS, all S3 storage integrations in an account share one IAM user. DESCRIBE INTEGRATION returns that user's ARN.

> **EXAM TRAP**
> Quote each blocked URL separately. Wrapping the whole STORAGE_BLOCKED_LOCATIONS value in quotes makes it invalid, and the setting is ignored.

Privileges for stages

Creating a stage with an integration needs USAGE on the integration. The role also needs CREATE STAGE on the schema. Loading or unloading through that stage needs only USAGE on the stage, not on the integration. Only an integration's owner can grant privileges on it.

Changing an integration

CREATE OR REPLACE on a storage integration breaks its stages. Stages link to the integration by a hidden ID, not by name. Setting ENABLED = FALSE blocks new stages and cuts existing stages off from their storage location.

EXAM TRAP

Revoking cloud access does not stop Snowflake at once. Cached temporary credentials can keep working for up to **60** minutes.

Cloud-specific settings

On AWS, set the same external ID in two places. STORAGE_AWS_EXTERNAL_ID on the integration must match the external ID in the trust policy of the IAM role you set up for it. Location prefixes are s3 for public regions, s3gov for government regions and s3china for China. An Azure integration authenticates to one tenant, so all its storage accounts must belong to it. For government or cross-region China storage, use CREDENTIALS on CREATE STAGE instead.

Validating a storage integration

SYSTEM$VALIDATE_STORAGE_INTEGRATION tests a path with real file actions. It can write, read, list or delete a file, or try all of them. The path must be in STORAGE_ALLOWED_LOCATIONS. Read fails if the file is missing, write fails if it already exists, and any failed action makes the status failure.

Finding API integrations

A system function finds integrations for a secrets provider. SYSTEM$GET_SECURITY_INTEGRATIONS_FOR_API_PROVIDER returns a JSON array of API_AUTHENTICATION integrations using WORKLOAD_IDENTITY_FEDERATION that your role can see. The provider is AWS_SECRETS_MANAGER, AZURE_KEY_VAULT or GCP_SECRET_MANAGER, in any case.

Openflow connectors

Openflow connectors ingest data from Kafka, Kinesis and MySQL. The Kafka connector writes topics into tables with Snowpipe Streaming High Performance. The MySQL connector soft-deletes rows through `_SNOWFLAKE_DELETED`. By default it replicates values up to **16** MB and fails a table on a larger value.

Kinesis and MySQL connector limits

Feature	Kinesis Data Streams	MySQL
Reads from	Only one stream per connector	MySQL or MariaDB
Scaling	No autoscaling	Single-node runtime: Min and Max nodes set to **1**
Formats or login	JSON by default; Avro and Protobuf need extra setup	Username and password only

Key takeaways

- Storage integrations let stages reach S3, GCS or Azure without explicit credentials.
- Creating a stage needs USAGE on the integration; using the stage needs only USAGE on the stage.
- CREATE OR REPLACE on a storage integration breaks the stages that reference it.
- Revoked cloud access can keep working for up to **60** minutes through cached credentials.

Sources: Option 1: Configure a Snowflake storage integration to access Amazon S3 | Snowflake Documentation (opens in a new tab) · CREATE STORAGE INTEGRATION | Snowflake Documentation (opens in a new tab) · Managing integrations in Snowsight | Snowflake Documentation (opens in a new tab) · Managing Snowflake integrations with Python | Snowflake Documentation (opens in a new tab) · Snowflake Openflow Connector for Kafka | Snowflake Documentation (opens in a new tab) · Openflow Connector for Amazon Kinesis Data Streams | Snowflake Documentation (opens in a new tab) · About Openflow Connector for MySQL | Snowflake Documentation (opens in a new tab) · ALTER STORAGE INTEGRATION | Snowflake Documentation (opens in a new tab)

Domain 4 | 21% of the exam | 15 min read

Performance Optimization, Querying, and Transformation

Overview

This domain is about 21% of the exam, roughly 21 of its 100 questions.

Every point below comes from Snowflake's official documentation; each section lists its sources.

Evaluate query performance

Start here

THINK FIRST

A query is slow or costly: which QUERY_HISTORY column, query insight or Account Usage view shows you why?

Why it matters: Questions describe a slow or costly query and ask which column, insight or view reveals the cause.

QUERY_HISTORY records each query's spilling, pruning and queuing. QUERY_INSIGHTS turns problems into labeled insights, one row each. QUERY_ATTRIBUTION_HISTORY shows what each query cost on a warehouse.

Key terms

Term	Meaning
Remote spillage	Data a query spilled to remote disk, shown in `bytes_spilled_to_remote_storage`.
Pruning	How many micro-partitions a query avoids reading; compare partitions scanned with the total.
Exploding join	A join problem flagged by `QUERY_INSIGHT_EXPLODING_JOIN` under the JOIN insight topic.
Queued overload	Time a query waited in the queue because the warehouse was overloaded.
Adaptive Warehouse	A warehouse kind other than a standard warehouse, with its own view for per-query credit usage.

Reading QUERY_HISTORY

QUERY_HISTORY shows spilling, pruning and queuing for each query. `bytes_spilled_to_remote_storage` shows data spilled to remote disk. Compare `partitions_scanned` with `partitions_total` to judge pruning. The view covers **365** days, while the Information Schema function covers only **7**.

> **EXAM TRAP**
> Not every query in QUERY_HISTORY also appears in ACCESS_HISTORY. When you query ACCESS_HISTORY, filter on `query_start_time` with a narrow range.

Overload and provisioning queuing

Two queue time columns show why a query waited. Overload means the current workload is too much for the warehouse. Provisioning means the warehouse was still being created, resumed or resized.

Two queue time columns compared

Feature	queued_overload_time	QUEUED_PROVISIONING_TIME
Why the query waits	The warehouse is overloaded by the current workload	Compute is provisioning after creation, resume or resize
Unit	Milliseconds	Milliseconds

Query insights

QUERY_INSIGHTS holds one row per insight produced for a query. Remote spillage and queued overload appear under the WAREHOUSE topic. Exploding joins appear under the JOIN topic, as `QUERY_INSIGHT_EXPLODING_JOIN` or `QUERY_INSIGHT_NESTED_EXPLODING_JOIN`.

Pruning history views

Pruning history views help you choose clustering keys. A good cluster key means fewer micro-partitions to scan for filters, joins and aggregates. TABLE_QUERY_PRUNING_HISTORY finds the tables queries access, and COLUMN_QUERY_PRUNING_HISTORY finds their often-used columns. Its ACCESS_TYPE column shows WHERE or JOIN use. TABLE_PRUNING_HISTORY compares pruning before and after Automatic Clustering or search optimization.

> **EXAM TRAP**
> COLUMN_QUERY_PRUNING_HISTORY reflects the plan actually executed, not your SQL text, so a HAVING clause can show as WHERE. It has no data for hybrid tables.

Query attribution

QUERY_ATTRIBUTION_HISTORY shows each query's warehouse compute cost. Use it for standard warehouses. Attributed credits leave out warehouse idle time. For an accelerated query, add `CREDITS_USED_QUERY_ACCELERATION` to `CREDITS_ATTRIBUTED_COMPUTE` for the total.

EXAM TRAP

Queries of about **100** ms or less never appear in QUERY_ATTRIBUTION_HISTORY. They are too short for per-query cost attribution.

Adaptive Warehouse metering

Adaptive Warehouse queries use QUERY_METERING_HISTORY instead. It gives per-query credits for **365** days, with one row per metering hour while a query runs. Charges can take up to **1** hour to appear. Roles granted USAGE_VIEWER or GOVERNANCE_VIEWER can see it.

Repeated statements

AGGREGATE_QUERY_HISTORY groups repeated statements by the minute. It holds data like QUERY_HISTORY, aggregated in one-minute intervals. It suits high-throughput workloads that repeat a few distinct statements, such as Unistore workloads. Queries that differ only in literal values can share one `QUERY_PARAMETERIZED_HASH`.

Latency at a glance

Account Usage views lag by different amounts. Check a view's latency before you decide that a recent query is missing from it.

Latency of the main query views

View	Latency	What it covers
QUERY_HISTORY	Up to **45** minutes	Each query, for **365** days
QUERY_INSIGHTS	Up to **90** minutes	One row per insight
QUERY_ATTRIBUTION_HISTORY	Up to **8** hours	Compute cost per query
TABLE_PRUNING_HISTORY	Up to **6** hours	Pruning for each table
COLUMN_QUERY_PRUNING_HISTORY	Up to **4** hours	Column access, kept for **1** year

Key takeaways

- Remote spilling shows in `bytes_spilled_to_remote_storage`; pruning shows in partitions scanned against the total.
- Queued overload means a busy warehouse; queued provisioning means compute was still starting or resizing.
- QUERY_ATTRIBUTION_HISTORY leaves out idle time and queries of about **100** ms or less.
- Account Usage QUERY_HISTORY keeps **365** days; the Information Schema function keeps **7**.

Sources: QUERY_HISTORY view | Snowflake Documentation (opens in a new tab) · QUERY_INSIGHTS view | Snowflake Documentation (opens in a new tab) · QUERY_ATTRIBUTION_HISTORY view | Snowflake Documentation (opens in a new tab) · TABLE_PRUNING_HISTORY view | Snowflake Documentation (opens in

a new tab) · QUERY_METERING_HISTORY view | Snowflake Documentation (opens in a new tab) · ACCESS_HISTORY view | Snowflake Documentation (opens in a new tab) · AGGREGATE_QUERY_HISTORY view | Snowflake Documentation (opens in a new tab) · Optimizing storage for performance | Snowflake Documentation (opens in a new tab)

Optimize query performance

Start here

THINK FIRST

A query runs slowly: when do you reach for query acceleration, search optimization, a clustering key or a materialized view?

Why it matters: Questions describe a slow query or table and ask which performance feature, setting, cost or limit fits it.

Each feature speeds up a different kind of query. Query acceleration offloads parts of large outlier queries to shared compute, and search optimization builds a search access path for point lookups. Clustering keys co-locate data in micro-partitions, and the optimizer can swap in a materialized view for its base table.

Key terms

Term	Meaning
Query acceleration service	Offloads portions of outlier queries to shared compute resources that the service provides.
Search access path	The persistent data structure that search optimization builds and maintains to speed up searches.
Clustering key	Columns or expressions chosen to co-locate a table's data in the same micro-partitions.
Point lookup	A selective query that finds just one row or a few distinct rows.
Analyzer	A setting named in the ALTER TABLE command for FULL_TEXT search optimization and again in the SEARCH function call.

Query acceleration

Query acceleration suits large scans with a selective filter or aggregation. It also helps large scans that insert or copy many new rows. Turn it on with `ENABLE_QUERY_ACCELERATION = TRUE` on CREATE WAREHOUSE or ALTER WAREHOUSE. Gen2 standard and multi-cluster warehouses have it on by default. Its total cost is the warehouse plus the service.

EXAM TRAP

Queries with nondeterministic functions such as SEQ or RANDOM are not eligible. A query with LIMIT also needs ORDER BY to qualify.

Checking eligibility

SYSTEM$ESTIMATE_QUERY_ACCELERATION tests whether one query is eligible. The QUERY_ACCELERATION_ELIGIBLE view finds queries and warehouses that might benefit. The two services can also work together: search optimization prunes micro-partitions first, then query acceleration offloads part of the remaining work.

Search optimization

Search optimization targets selective point lookups. It works best when a filtered column has about **100,000** or more distinct values. Queries that already run in under a second rarely gain much. The optimizer decides when to use it, and a query needs only SELECT on the table.

Example: Adds search optimization to a whole table.

```
ALTER TABLE my_table ADD SEARCH OPTIMIZATION;
```

- `ADD SEARCH OPTIMIZATION`: with no ON clause, columns added later also get EQUALITY

What search optimization skips

External, hybrid and temporary tables are not supported. Queries that use Time Travel gain nothing, because it works only on active data. FLOAT, DECFLOAT and GEOMETRY values do not benefit, and neither do most predicates that cast the column's values.

EXAM TRAP

FULL_TEXT search optimization serves the SEARCH function, and its ALTER TABLE command names an analyzer. The SEARCH call names an analyzer too. If the two differ, the search access path is not used.

Search optimization costs

Maintenance needs no warehouse of yours, but it is not free. Its storage and compute are charged. SYSTEM$ESTIMATE_SEARCH_OPTIMIZATION_COSTS gives an estimate, but real costs can differ by up to **50%**. The access path is usually about **1/4** of the table's size. Enabling specific columns, not the whole table, is the usual best practice.

EXAM TRAP

On a high-churn table, adding Automatic Clustering to search optimization can raise maintenance costs above search optimization alone.

Clustering keys

Keep a clustering key to 3 or 4 columns or expressions. More columns tend to add cost faster than benefit. Clustering pays off most on tables that are queried often and change rarely.

Clustering compared with search optimization

Feature	Clustering	Search optimization
Best for	Range queries and inequality filters	Selective point lookups
How it helps	Co-locates data in the same micro-partitions	Builds and maintains a search access path
Scope on a table	One clustering key per table	Best enabled for specific columns

Materialized views

A materialized view can query only one table. Joins, including self-joins, are not supported. The optimizer can rewrite base-table queries to use the view automatically. Materialized views can also speed up queries on external tables and let you cluster the same data on a different key.

EXAM TRAP

A materialized view that is behind its base table still returns up-to-date results. The query may just take a little longer.

Key takeaways

- Query acceleration offloads parts of large outlier queries to shared compute.
- Search optimization speeds up point lookups on columns with about **100,000** or more distinct values.
- Keep a clustering key to **3** or **4** columns or expressions.
- A materialized view queries a single table and supports no joins.

Sources: Using the Query Acceleration Service (QAS) | Snowflake Documentation (opens in a new tab) · Search optimization service | Snowflake Documentation (opens in a new tab) · Clustering Keys & Clustered Tables | Snowflake Documentation (opens in a new tab) · Working with Materialized Views | Snowflake Documentation (opens in a new tab) · Enabling and disabling search optimization | Snowflake Documentation (opens in a new tab) · Tutorial: Improve Workload Performance with the Query Acceleration Service | Snowflake Documentation (opens in a new tab) · Optimizing query performance | Snowflake Documentation (opens in a new tab) · Identifying queries that can benefit from search optimization | Snowflake Documentation (opens in a new tab)

Use Snowflake caching

Start here

THINK FIRST

When does Snowflake reuse a cached result, and when does a warehouse setting throw a cache away?

Why it matters: Questions describe a query or a warehouse setting and ask whether a cache is reused, lost or worth keeping.

Snowflake persists query results for reuse, and each running warehouse caches table data. Result reuse needs an exactly matching query. The warehouse cache is dropped when the warehouse suspends, so

auto-suspend trades saved credits against speed.

Key terms

Term	Meaning
Query result cache	Persisted results Snowflake can reuse; each reuse resets **24** hours, up to **31** days.
Warehouse cache	Table data a running warehouse keeps for later queries on the same warehouse.
USE_CACHED_RESULT	The parameter that overrides result reuse at the account, user or session level.
RESULT_SCAN	A table function that returns an earlier query's results as a table.
Interactive warehouse	A warehouse you attach interactive tables to, up to **100**, so it can warm a cache for those tables.

When results are reused

Result reuse needs the new query to match the old one exactly. Even a change in letter case or a table alias stops full reuse. Reuse is on by default, and USE_CACHED_RESULT can override it for an account, user or session.

EXAM TRAP

Functions such as UUID_STRING, RANDOM and RANDSTR block reuse. So do external functions and queries on hybrid tables.

How long results last

Persisted results expire after 24 hours. Each reuse resets the **24** hours, up to **31** days from the first run. For results over **100** KB, the access token expires after **6** hours. A cached SHOW result is reused only by the role that produced it.

Working with RESULT_SCAN

RESULT_SCAN turns an earlier result into a table you can query. With no argument, it reads the most recent command. It works on results from any of your sessions within **24** hours, a period you cannot adjust. Add ORDER BY when row order matters.

EXAM TRAP

Even ACCOUNTADMIN cannot use RESULT_SCAN on another user's results. For manual queries, access belongs to the user who ran them, not the role.

Metadata without a warehouse

Some metadata commands, such as SHOW TABLES, need no warehouse. With no warehouse involved, no warehouse charges apply. To post-process SHOW output with RESULT_SCAN, double-quote the column

names, because they are lowercase.

The warehouse cache

A running warehouse caches table data for its own queries. Later queries on the same warehouse can read the cache instead of the tables. A larger warehouse has a larger cache. Suspending drops it, and after a resume some queries run slower until it is rebuilt.

EXAM TRAP

Resizing down also costs cache. The removed compute resources take their cache with them, much like a suspend.

Choosing auto-suspend

Set auto-suspend by how much the workload needs the cache. For tasks, suspend immediately. AUTO_SUSPEND is set in seconds and defaults to **600**, which is **10** minutes. A value of **0** or NULL means the warehouse never suspends.

Suggested auto-suspend by workload

Workload	Auto-suspend	Cache value
BI and SELECT queries	At least **10** minutes	High: it keeps the cache for users
DevOps, DataOps and Data Science	About **5** minutes	Lower: queries are ad-hoc and unique

EXAM TRAP

AUTO_SUSPEND is not precise. The background process that suspends warehouses runs only about every **30** seconds.

Interactive warehouse caching

ALTER WAREHOUSE ... ADD TABLES starts cache warming. It attaches interactive tables to an interactive warehouse, up to **100** tables. A standard warehouse cache only rebuilds as queries run; this warms the cache for the chosen tables. DROP TABLES stops using the cache for those tables but does not drop the tables.

Key takeaways

- Result reuse needs an exact query match; even a case or alias change prevents full reuse.
- Persisted results last **24** hours, reset by each reuse up to **31** days.
- Suspending or downsizing a warehouse drops its cache.
- Keep BI warehouses running at least **10** minutes before suspending; ad-hoc ones about **5**.

Sources: DESCRIBE RESULT | Snowflake Documentation (opens in a new tab) · Optimizing the warehouse cache | Snowflake Documentation (opens in a new tab) · Using Persisted Query Results | Snowflake Documentation (opens in a new tab) · Warehouse considerations | Snowflake Documentation (opens in a new tab) · ALTER WAREHOUSE | Snowflake Documentation (opens in a new tab) · RESULT_SCAN | Snowflake

Documentation (opens in a new tab)

Perform data transformation techniques

Start here

THINK FIRST

For a given data shape and result, how do you choose between aggregate and window functions, and handle VARIANT data or staged files?

Why it matters: Questions show a query, a data shape or a result and ask which function, clause or setting produces it.

An aggregate function returns one row per group, while a window function returns one row per input row. Semi-structured data sits in VARIANT, OBJECT and ARRAY columns. Unstructured files stay on a stage and are reached through scoped, file or pre-signed URLs, or FILE references.

Key terms

Term	Meaning
Window function	An analytic function that returns one row per input row, used for running totals, moving averages and rankings.
OVER clause	Holds a window function's PARTITION BY, ORDER BY and window frame specification.
VARIANT null	A real null value in semi-structured data that equals itself, unlike a missing SQL NULL.
Scoped URL	A temporary URL to a staged file, made with BUILD_SCOPED_FILE_URL and usable only by its creator.

Aggregate or window

Aggregates collapse a group; window functions keep every row. Use a window function for running totals, moving averages and rankings. An empty `OVER()` clause is valid.

Scalar, aggregate and window functions compared

Feature	Scalar	Aggregate	Window
One result per	Invocation, usually each row	Group of rows	Input row in a partition
Output	One value	One row per group	One row per input row

Ordering inside OVER

ORDER BY in OVER orders the window, not the query output. Ranking functions such as RANK and NTILE require it. Frames for running totals or moving averages need it too, or the frame is meaningless. Declare

window frames explicitly instead of relying on an implied one.

EXAM TRAP

`ORDER BY 2` inside OVER does not mean the second column. A number there is treated as the constant **2**.

Window frames

In a RANGE frame, CURRENT ROW covers every row with the same ORDER BY value. A RANGE BETWEEN frame with explicit offsets must have exactly one ORDER BY expression. Correlated columns inside window functions, for example in PARTITION BY within a correlated subquery, are not supported.

Working with VARIANT

A VARIANT value holds up to 128 MB of uncompressed data. TYPEOF returns a stored value's native type. Dates and timestamps loaded from JSON are stored as strings, so they can be slower and larger than typed columns. Strings and dates come back in double quotes until you cast them.

EXAM TRAP

SQL NULL in a VARIANT means a missing value, while a VARIANT null is a real value. IS_NULL_VALUE tests JSON nulls, not SQL NULL.

Arrays and objects

Array indexes start at 0. Reading past the end of an array returns NULL, not an error. ARRAY_SLICE returns elements from the start index up to, but not including, the end index. In an OBJECT, each key is a VARCHAR and each value is a VARIANT.

Search optimization on nested data

Semi-structured and structured columns need the ON clause. Table-level search optimization covers neither kind. It can speed up point lookups that cast a semi-structured column to VARCHAR. Substring search uses only tokens of at least **5** characters, so `LIKE '%TEST%'` gains nothing.

EXAM TRAP

On a VARIANT column, enabling an element also covers its nested elements. On a structured column, nested elements are not covered.

Unstructured files

Unstructured files must sit on an internal or external stage. A FILE value stores only a reference to the file, and it does not change when the file does.

Three ways to reach a staged file

Feature	Scoped URL	File URL	Pre-signed URL
Who can use it	Only the user who generated it	Roles with enough privileges on the stage	Any user or application
Access needed	No stage privileges; expires after **24** hours	USAGE on an external stage, READ on an internal one	No Snowflake login or token

EXAM TRAP

Internal stages encrypt files client-side by default, and the type cannot change later. If you want unstructured data access, consider server-side encryption at stage creation.

Key takeaways

- Aggregates return one row per group; window functions return one row per input row.
- ORDER BY inside OVER orders only the window, and RANK and NTILE require it.
- Array indexes start at **0**, and reading past the end returns NULL.
- Search optimization on VARIANT, OBJECT or ARRAY columns needs the ON clause.

Sources: Analyzing data with window functions | Snowflake Documentation (opens in a new tab) · Semi-structured data types | Snowflake Documentation (opens in a new tab) · Speeding up queries of semi-structured data with search optimization | Snowflake Documentation (opens in a new tab) · Unstructured data types | Snowflake Documentation (opens in a new tab) · Introduction to unstructured data | Snowflake Documentation (opens in a new tab) · REST API for unstructured data support | Snowflake Documentation (opens in a new tab) · Window functions | Snowflake Documentation (opens in a new tab) · Scalar functions | Snowflake Documentation (opens in a new tab)

Domain 5 | 10% of the exam | 13 min read

Data Collaboration

Overview

This domain is about 10% of the exam, roughly 10 of its 100 questions.

Every point below comes from Snowflake's official documentation; each section lists its sources.

Explain data collaboration and protection

Start here

THINK FIRST

How do Time Travel, Fail-safe, cloning, sharing and replication each protect or share data, and when would you reach for each one?

Why it matters: Questions describe a recovery, sharing or disaster-recovery need and ask which feature, limit or command fits it.

Time Travel and Fail-safe protect an object's history, one after the other. Secure Data Sharing gives other accounts read-only access without copying data. Replication and failover keep copies in other accounts so you can switch over.

Key terms

Term	Meaning
Retention period	The length of an object's Time Travel window: **1 day** as standard, **0** to **90 days** for permanent objects.
Fail-safe	A fixed **7-day** window after Time Travel when only Snowflake can recover data.
Share	An object a provider creates to give consumer accounts read-only access to its data.
Primary database	The database where all changes run; secondary copies in other accounts refresh from it.
Failover group	A group of objects replicated together that a target account can promote to primary.

Time Travel retention

Every account gets 1 day of Time Travel by default. Permanent databases, schemas and tables can keep history for **0** to **90 days** on Enterprise Edition and higher. You cannot turn Time Travel off for an account. An ACCOUNTADMIN can still set `DATA_RETENTION_TIME_IN_DAYS` to **0** at account level.

EXAM TRAP

Setting an object's retention to **0** does not always remove its history. If `MIN_DATA_RETENTION_TIME_IN_DAYS` is set on the account, the larger of the two values applies.

Querying and cloning history

AT or BEFORE picks a point in time three ways. Use `TIMESTAMP` for a moment, `OFFSET` for a number of seconds back, or `STATEMENT` with a query ID. You can also clone an object as it was at that point. Such a clone leaves out external tables and internal stages.

EXAM TRAP

A historical clone of a schema will not bring back its internal stages or external tables; those object types are skipped.

Fail-safe after Time Travel

Fail-safe adds 7 days that you cannot change. The window opens right after the Time Travel period ends. It exists for Snowflake to recover data lost in extreme operational failures. A recovery can take several hours or several days.

Time Travel and Fail-safe compared

Feature	Time Travel	Fail-safe
Length	**1 day** by default, up to **90 days** for permanent objects	Fixed **7 days**, not configurable
Who recovers data	You, with AT or BEFORE queries	Only Snowflake
When it is available	Enabled automatically for all accounts	Only after the Time Travel period ends

EXAM TRAP

Fail-safe is not a self-service backup. You cannot recover from it yourself; only Snowflake uses it after extreme failures, and recovery is slow.

Secure Data Sharing

Sharing copies no data between accounts. Shared data adds nothing to the consumer's storage bill. The consumer pays only for the warehouses that query it. A consumer can accept many shares but creates just **1** database from each share. Creating a share needs ACCOUNTADMIN or the global CREATE SHARE privilege.

EXAM TRAP

Consumers cannot change shared objects. They cannot insert rows, modify tables or drop objects, because everything shared is read-only.

Protecting shared data

Share secure views or secure UDFs, not raw tables. Snowflake strongly recommends this so sensitive data stays hidden from consumer accounts. Before sharing, test what a consumer will see. Set the session parameter `SIMULATED_DATA_SHARING_CONSUMER` to that consumer's account, then query the secure view.

Primary and secondary databases

All changes run on the primary; secondaries are read-only. You can enable replication on any permanent or transient database, which makes it the primary. Secondary copies can live in other accounts of the same organization, in other regions or clouds. Each refresh brings a snapshot of data and DDL. With scheduled refreshes, a secondary lags at most **2x** the refresh interval.

What database replication skips

Stages, pipes and grants need account replication. Database replication copies data and object definitions, but some things stay behind. A database created from a share cannot be replicated at all. Replicating from Business Critical to a lower edition is blocked unless you add `IGNORE EDITION CHECK`.

Database replication and account replication compared

Feature	Database replication	Account replication
Stages and pipes	Not supported	Supported
Privilege grants, including future grants	Not replicated	Needed to replicate them

EXAM TRAP

A secondary database does not arrive with the primary's grants. Privileges on its objects, including future grants, are not copied with the data.

Failing over to another account

Run the promote command in the target account. Replication and failover groups copy a set of objects with point-in-time consistency. To fail over, a role with the FAILOVER privilege signs in to the target account. Snowflake blocks the promotion while a refresh is running. Client Redirect gives clients one connection URL that can point to another account.

Example: Promotes a secondary failover group to primary, run while signed in to the target account.

```
ALTER FAILOVER GROUP my_fg PRIMARY;
```

- `PRIMARY`: makes this target account serve as the source account

EXAM TRAP

Failover suspends scheduled refreshes on every secondary failover group. They stay off until RESUME runs in each target account.

Key takeaways

- Time Travel defaults to **1 day** and reaches **90 days** for permanent objects; Fail-safe adds a fixed **7 days**.
- Only Snowflake can recover data from Fail-safe, and it can take hours or days.
- Shared objects are read-only and cost the consumer only query compute.
- Database replication skips stages, pipes and grants; account replication covers them.
- After a failover, resume refreshes in each target account with `ALTER FAILOVER GROUP … RESUME`.

Sources: Understanding and viewing Fail-safe | Snowflake Documentation (opens in a new tab) · About Secure Data Sharing | Snowflake Documentation (opens in a new tab) · Introduction to business continuity & disaster recovery | Snowflake Documentation (opens in a new tab) · Understanding & using Time Travel | Snowflake Documentation (opens in a new tab) · Use secure objects to control data access | Snowflake Documentation (opens in a new tab) · Introduction to database replication across multiple accounts | Snowflake Documentation (opens in a new tab) · Failing over account objects | Snowflake Documentation (opens in a new tab)

Explain Snowflake's data sharing capabilities

Start here

THINK FIRST

When would a provider choose a direct share, a listing, a reader account or a clean room to share data?

Why it matters: Questions describe a sharing need and ask which account type, share method, privilege or limit applies to it.

A provider places database objects in a share, and consumer accounts in the same region query them. Reader accounts are consumers the provider creates and pays for. Listings reach other regions, and clean room partners return aggregated results without directly querying raw data.

Key terms

Term	Meaning
Provider	The account that shares its data with other accounts.
Consumer	The account that receives and queries the shared data.
Share	A named object that holds everything needed to share a database.
Reader account	An account the provider creates, owns and pays for, which reads data from that provider only.

Term	Meaning
Database role	A role inside a database, granted to a share, that exposes a subset of its objects.

Providers, consumers and shares

A provider shares data with consumers through a share. The share is a named object that carries everything needed to expose a database. When the provider adds an object to the share, every consumer can use it at once.

EXAM TRAP

Adding a table to a share needs no refresh or new share: all consumers see it immediately.

Database roles or direct grants

Database roles let consumers give groups different subsets of a share. Without them, a consumer grants the single `IMPORTED PRIVILEGES` privilege, which opens every shared object. Direct grants can combine several databases, but only from one account. A shared database role takes no future grants.

Two ways to put objects into a share

Feature	Database roles	Privileges granted directly
Consumer access	Groups can get different subsets of objects	`IMPORTED PRIVILEGES` gives access to every object
Several databases in one share	Not possible: the `REFERENCE_USAGE` privilege that multi-database shares need cannot go to a database role	Allowed when the databases belong to the same account

EXAM TRAP

A share that spans several databases cannot use database roles, because a database role cannot receive the `REFERENCE_USAGE` privilege.

Creating reader accounts

The provider creates, owns and pays for each reader account. Use `ACCOUNTADMIN` or a role with the global `CREATE ACCOUNT` privilege. A reader account can consume data only from the provider that created it. By default, a provider can create **75** reader accounts in total.

Example: Creates a reader account with its first administrator user.

```
USE ROLE ACCOUNTADMIN;
CREATE MANAGED ACCOUNT my_reader_acct
  ADMIN_NAME = my_admin,
  ADMIN_PASSWORD = 'my_password',
  TYPE = READER;
```

- `TYPE = READER`: makes the managed account a reader account

Inside a reader account

A new reader account has one admin user and no data. The admin creates a database from each share, and queries need a virtual warehouse. Its credits are billed to the provider with no monthly cap. Set a resource monitor to limit them.

Example: Turns a share from the provider into a database the reader account can query.

```
CREATE DATABASE my_shared_db
  FROM SHARE my_provider_acct.my_share;
```

EXAM TRAP

A reader account cannot load or change data: `INSERT`, `UPDATE`, `DELETE`, `MERGE` and `COPY INTO <table>` are not allowed there.

Direct shares, listings and resharing

A direct share reaches only accounts in the same region. To reach another region or cloud platform, publish a listing that uses cross-cloud auto-fulfillment. Resharing works differently for the two methods.

Direct shares and listings compared

Feature	Direct share	Listing
Reach	Accounts in the same region only	Other regions and clouds through cross-cloud auto-fulfillment
Allowing resharing	Possible for views, tables and dynamic tables only	Set `resharing.enabled` to `true` in the manifest

EXAM TRAP

A direct share cannot reach an account in another region, even on the same cloud platform. That consumer needs a listing with cross-cloud auto-fulfillment.

Data clean rooms

Collaborators return aggregated results but cannot query raw data directly. Each role has its own edition needs, and the party that runs an analysis pays for it. Trial accounts cannot use Snowflake Data Clean Rooms.

EXAM TRAP

Only data providers need Enterprise Edition. Standard Edition is enough for clean room owners and analysis runners.

Key takeaways

- A direct share stays in one region; a listing that uses cross-cloud auto-fulfillment reaches other regions and clouds.
- The provider pays every reader account credit, so put a resource monitor on its warehouses.
- Reader accounts cannot load or change data with `INSERT`, `UPDATE`, `DELETE`, `MERGE` or `COPY INTO <table>`.
- Database roles split a share into subsets, but they cannot span databases or use future grants.
- Clean room partners can return aggregated results but cannot directly query the raw data, and the analysis runner pays for each analysis.

Sources: About Secure Data Sharing | Snowflake Documentation (opens in a new tab) · Manage reader accounts | Snowflake Documentation (opens in a new tab) · About Snowflake Data Clean Rooms | Snowflake Documentation (opens in a new tab) · Share secure database objects | Snowflake Documentation (opens in a new tab) · Configure a reader account | Snowflake Documentation (opens in a new tab) · Using resharing as a provider | Snowflake Documentation (opens in a new tab) · Overview of Snowflake Data Clean Rooms | Snowflake Documentation (opens in a new tab)

Share data using the Snowflake Marketplace and listings

Start here

THINK FIRST

When would a provider share through a private listing instead of a public Snowflake Marketplace listing?

Why it matters: Questions describe a sharing goal and ask which listing type, app component or resharing rule fits it.

A listing shares an app with consumers, either free or paid. Marketplace listings reach many consumers at once, while private listings go to specific consumers. Consumers can discover and install Native Apps from either kind.

Key terms

Term	Meaning
Listing	A way to share an application with consumers; a listing can be either free or paid.
Private listing	A listing sent straight to chosen Snowflake accounts in any region the Native App Framework supports.
Application package	The provider object holding an app's data references, code, manifest file and setup script.
Setup script	SQL that runs on each install or upgrade of the app; the manifest gives its location.

Term	Meaning
Resharing	A provider option that lets consumers pass an incoming listing on to other accounts.

Marketplace versus private listings

Marketplace listings reach many consumers; private ones target chosen accounts. One Marketplace listing replaces separate sharing relationships with each consumer. A private listing goes directly to a named account, in any region the Native App Framework supports. Consumers can discover and install apps from either kind.

Snowflake Marketplace and private listings compared

Feature	Snowflake Marketplace listing	Private listing
Audience	Many consumers at the same time	Specific consumer accounts the provider picks
Pricing	Free or paid	Free or paid
Native App install	Consumers can discover and install apps	Consumers can install apps shared to them

EXAM TRAP

A private listing is not limited to the provider's own region. It can reach accounts in any region the Native App Framework supports.

What a Native App contains

An application package needs a manifest file and a setup script. The package can also reference data content and external code files. Its logic can be a Streamlit app, stored procedures, or functions built with Snowpark API, JavaScript or SQL. On install, Snowflake creates the app object in the consumer account and runs the setup script.

Manifest and setup script rules

The manifest must be named `manifest.yml` and sit at the stage root. Place it at the root of the directory on the named stage that holds the other app files. It points to the setup script, which runs on every install or upgrade. Using `manifest_version: 2` turns on automated granting of privileges.

EXAM TRAP

The manifest file name is fixed. A file named `manifest.yaml` or `app.yml` does not meet the rule that it must be `manifest.yml`.

Consumer controls and resources

Consumers can limit which objects an installed app creates. They do this with feature policies. An optional `marketplace.yml` file declares needed resources, such as compute pools and external connections, and these show in the listing in Snowsight. An app with containers needs `CREATE COMPUTE POOL` unless the consumer creates the pool manually.

Trial listings

When a trial ends, Snowflake suspends the app unless the consumer converts it. Converting means moving the app to the full listing. Data written inside a suspended app is kept, as long as the consumer does not delete the app.

> **EXAM TRAP**
> A trial ending does not wipe the app's data. The data stays until the consumer deletes the app.

Resharing listings

If the provider allows it, consumers can reshare incoming listings. They can reshare within one region or across regions. Resharing needs the same roles and privileges as creating a listing. The default enforcement mode is `RESHARER`, so governance policies are evaluated in the resharer's account.

> **EXAM TRAP**
> The provider does not pay when a resharer sends a listing to another region. Auto-fulfillment replication costs are charged to the resharer.

Key takeaways

- Marketplace listings reach many consumers at once; private listings go to chosen accounts in any supported region.
- Every application package needs a setup script and a `manifest.yml` file at the stage root.
- An expired trial suspends the app but keeps its data until the consumer deletes the app.
- Cross-region reshares are paid for by the resharer, and policies default to the resharer's context.

Sources: About the Snowflake Native App Framework | Snowflake Documentation (opens in a new tab) · Use and manage Snowflake Native Apps as a consumer | Snowflake Documentation (opens in a new tab) · Resharing listings | Snowflake Documentation (opens in a new tab) · Create the manifest file for an app | Snowflake Documentation (opens in a new tab) · Specify the resources required by an app | Snowflake Documentation (opens in a new tab)

MockCert

Take a SnowPro Core mock exam the way the real one feels

5 timed mock exams · 100 questions in 115 minutes each · one is free

- Free to start: study notes, explained quizzes, flashcards and the first mock exam
- Timed mock exams built to the official exam guide
- A verifiable Certificate of Assessment when you pass a premium mock

Explore SnowPro Core on MockCert



Example MockCert certificate. Earn yours by passing a premium mock exam.

Open it on your phone

mockcert.com/snowpro-core/

Scan the code with your phone's camera.