Snowflake Builders Blog: …

Snowflake Access Control, Let make it easy for all

Rajiv Gupta Follow 8 min read · Dec 18, 2022

162 1

Photo by Rahul Pande on Unsplash

In this blog we are going to break down the most neglected topic in any database viz. Access Control. The majority of person don't like this as it’s always a difficult to understand the implementation of RBAC than to understand the concept. Concept wise, it's not that difficult and seems easy, but when you start implementing access control setup soon it grows to be very complex and if someone has not taken care of this properly it becomes worse. My attempt from this blog is to clarify the all the terminology and try to relate them to more user-friendly terms so that we can memorize the stuff without getting confused. This is how I started exploring Snowflake, and it is still helping me when I start looking at complex setups. So let’s start without wasting any time.

. . .

How much relevant this topic is with respect to certification exam?

This topic is very important, and you may expect a couple of question from this topic in SnowPro , SnowPro Advance Architect and SnowPro Advance Admin exam.

. . .

What is an Access Control Framework?

Snowflake’s approach to access control combines aspects from both of the following models:

- **Discretionary Access Control (DAC):**

Snowflake defines this as “Each object has an owner, who can in turn grant access to that object”.

```
My way of learning "Take this concept as you are a builder,
and you are making a huge building in some renowned spot.
As you had put money into that project, you are the owner of that project.
Hence, you have full ownership to sell, rent or keep all flat at your own
discretionary. Now if you correlate this term with any object creation
in Snowflake, then it will be clear to you what Snowflake is saying in the
above definition."
```

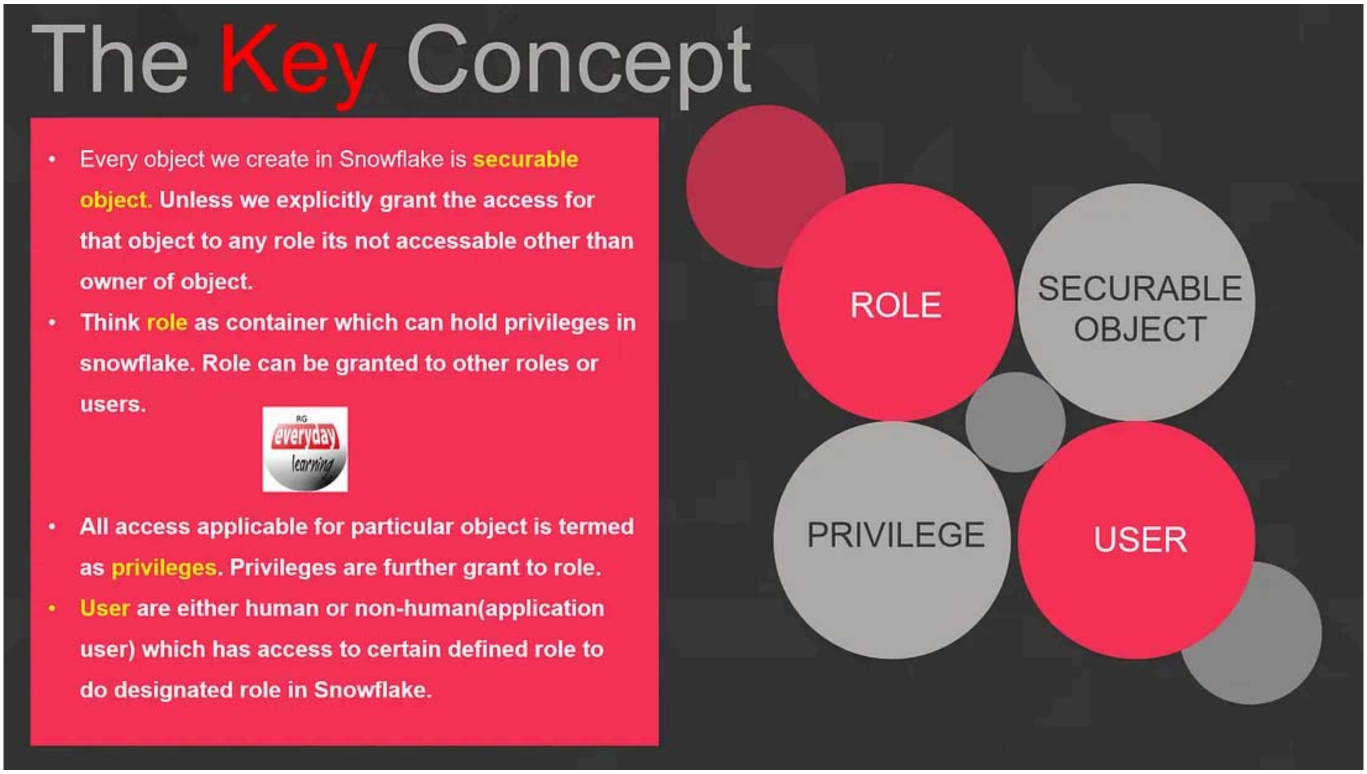
- **Role-Based Access Control (RBAC):**

Snowflake defines this as “Access privileges are assigned to roles, which are in turn assigned to users”.

```
My way of learning "Now just think that you went to above builder and
showed your intrest to lease a flat on rent. Builder agrees to lease you one
flat in his building on agreed rent. Now being a rightfull owner
(till the lease period is valid) of that flat you have
got rights to do certain things for your flat but you cant do anything for
any other flat in building. You have access to your flat not others.
Now if you correlate this term with any role/privilege grant in Snowflake,
then it will be clear to you what Snowflake is saying in the
above definition."
```

. . .

Some of the key concept to understand access control in Snowflake.



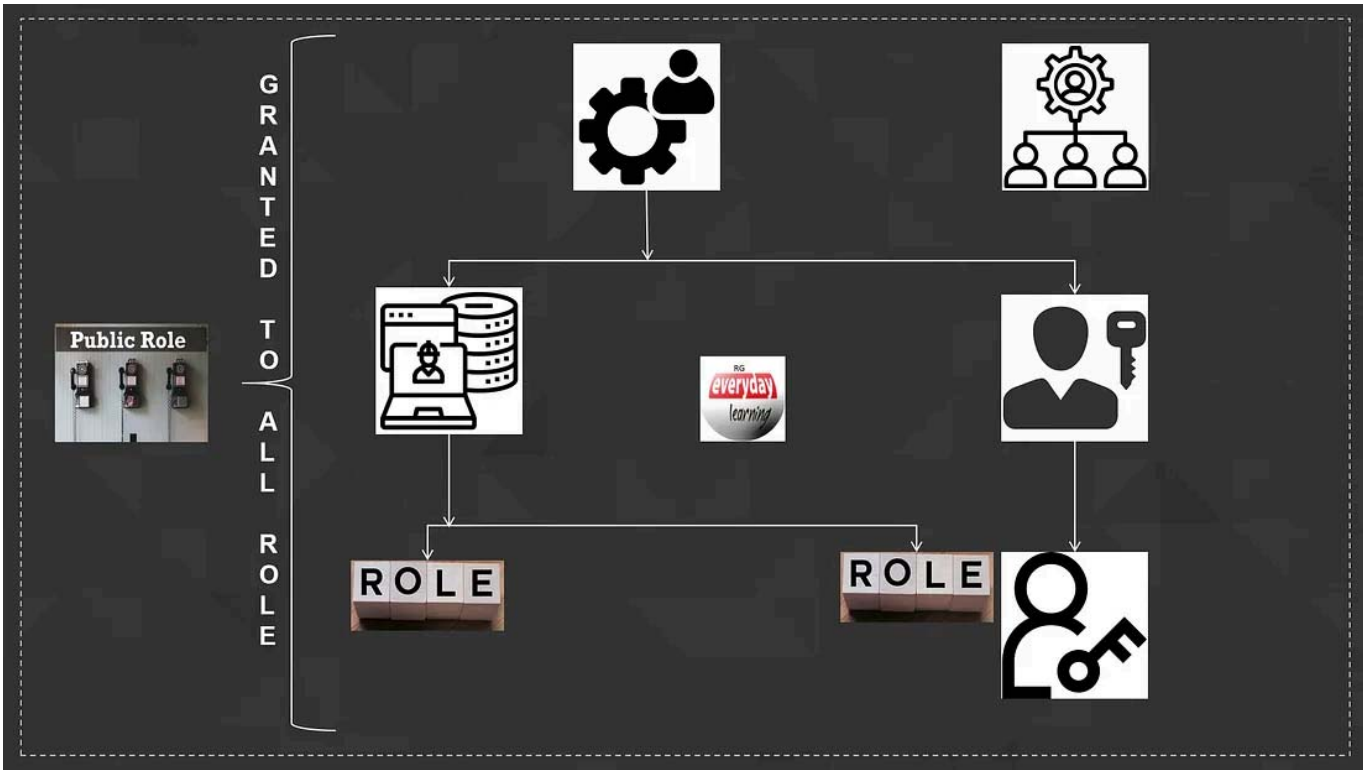
The Key Concept

- Every object we create in Snowflake is securable object. Unless we explicitly grant the access for that object to any role its not accessable other than owner of object.
- Think role as container which can hold privileges in snowflake. Role can be granted to other roles or users.
- All access applicable for particular object is termed as privileges. Privileges are further grant to role.
- User are either human or non-human(application user) which has access to certain defined role to do designated role in Snowflake.

. . .

What are predefined roles in Snowflake?

When a new Snowflake account is provisioned in Snowflake, a few predefined standard roles has been created automatically. We can’t drop this role. Neither we can revoke the privileges pre-granted to this role, but we can revoke the custom grant if we have granted post provisioning the account. Generally it is recommended to avoid giving any direct privilege to standard role.



Role Hierarchy

Let's understand them one by one.

ORGADMIN:

ORGADMIN

This role is relatively new as compared to other standard role. Last year this role has been introduced with new feature called Organization. This role is not aligned in any hierarchy to the existing standard role. This standard role is specifically meant for maintaining all organizational activity like below:

1. Create a new account under organization.
2. List all accounts in Organization.
3. List all enabled region in account.
4. Access all organizational usage information.

ACCOUNTADMIN:

ACCOUNTADMIN

This role is the highest level role in any account which is responsible to manage all the things in account. This role is specifically meant for Admin people who manage Snowflake account in any organization. Always tries to have access to this role to 2 people at minimum so that they can unlock each other account in case of one account is locked for any reason else unlocking may take time if snowflake support route is taken. This role encapsulates 2 standard roles in hierarchy i.e. SYSADMIN & SECURITYADMIN. This means that it also inherits the privilege of below 2 roles in accountadmin role.

SYSADMIN:

SYSADMIN

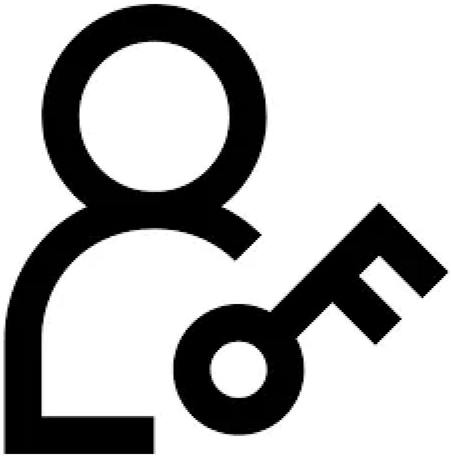
This role is designed to create warehouse and database(and other objects) in an account in Snowflake. It is also recommended to create role hierarchy and ultimately assign all custom role to SYSADMIN role so that it is managed by high level role. If any custom role is not tagged to sysadmin role than it remain unmanaged role and even ACCOUNTADMIN can't manage any object created by that custom role.

SECURITYADMIN:

SECURITYADMIN

This role is designed to create , monitor & manage any user or role in snowflake. This standard role has global manage grant privilege. This role also inherits the **USERADMIN** standard role privilege in hierarchy.

USERADMIN:



USERADMIN

This role is designed to specifically create user and role in account. This role is inherited by SECURITYADMIN role in hierarchy. This role can also manage the role and user created by this role or get the ownership via transfer of ownership.

PUBLIC:

PUBLIC role is the standard role which is specifically assigned to all the user created in snowflake account by default. This role is also granted to all role in snowflake. As the name suggests, this role is open to all and can be used for use cases where you don't want any kind of security and want to make access to certain object as publically accessible in snowflake.

. . .

What is custom role in Snowflake?

Any role other than the standard role created in snowflake is a custom account level role in snowflake. All custom role satisfies either of the below criteria . All custom roles have to be tagged under SYSADMIN role to make it managed role in any snowflake account.

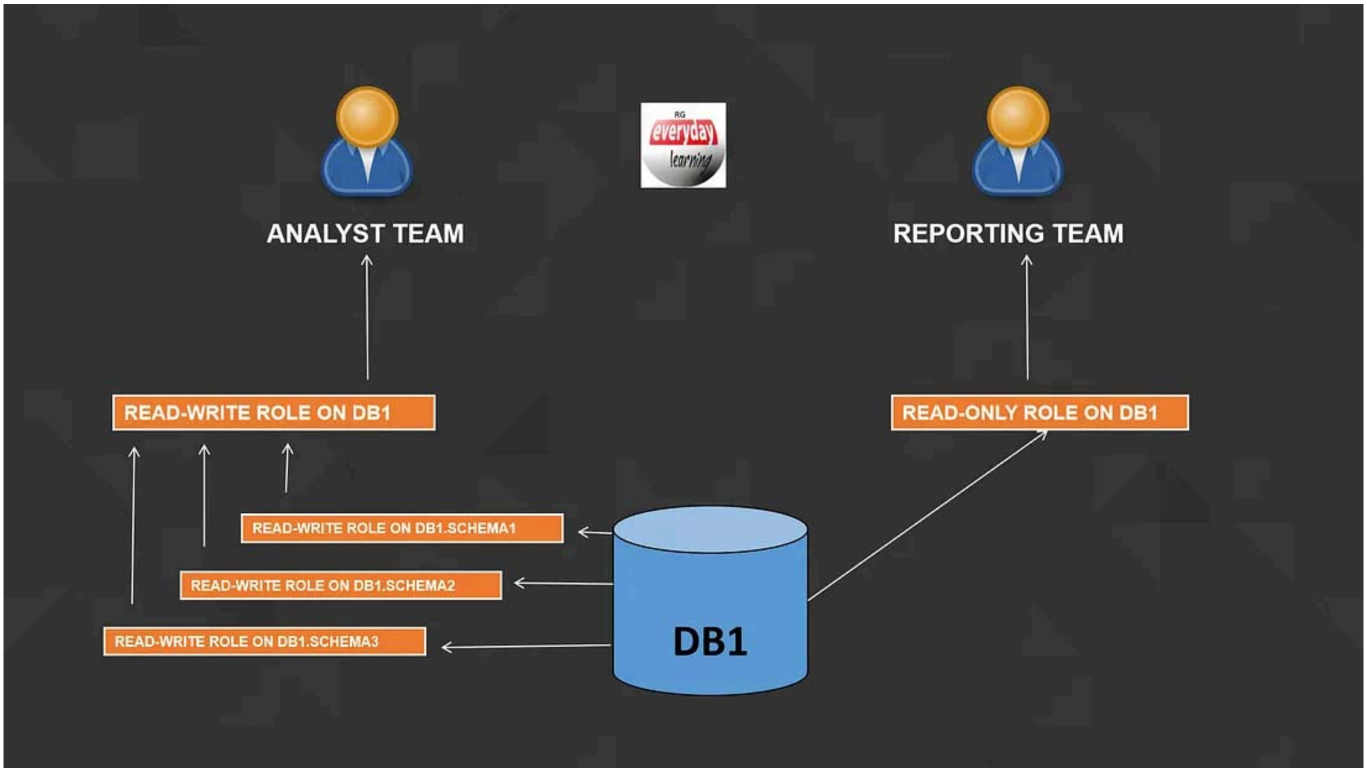
1. Access level role — Specifically used in snowflake to design the access hierarchy so that RBAC can be managed efficiently in snowflake e.g. Read-only role, Read-write role etc.
2. Functional role — Specifically used to cater the business needs of the application. All functional role inherit one or multiple access level role fulfill certain business needs e.g. Analyst, Developer, Testing etc.

. . .

What is privilege inheritance in Snowflake ?

Privilege inheritance is very easy to understand with the below diagram. You can see that certain roles are assigned to other roles in a hierarchical manner which is basically helping higher level role to inherit the privilege from lower level role.

This is just **opposite hierarchy** to how we inherit the heritage privilege to from our grandparents. Like your great-grandfather has left some heritage for your father and your father had transferred the same privilege to you, and you will share with your children and hence forth. This is the same but in **reverse order**. Here a lower level role is inherited by a higher level role to create a role hierarchy. At the same time they also inherit the privileges and access of lower level role in natural form.


ANALYST TEAM
REPORTING TEAM
READ-WRITE ROLE ON DB1
READ-ONLY ROLE ON DB1
READ-WRITE ROLE ON DB1.SCHEMA1
READ-WRITE ROLE ON DB1.SCHEMA2
READ-WRITE ROLE ON DB1.SCHEMA3
DB1

Privilege Inheritance

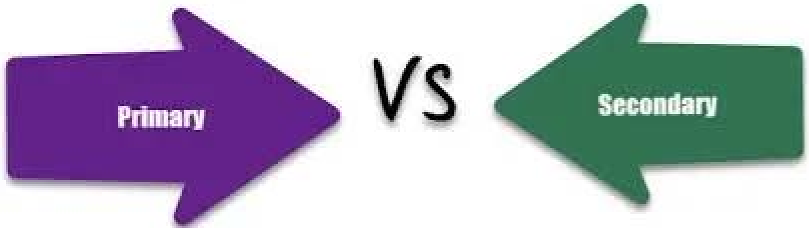
. . .

What is primary role and secondary role in snowflake?

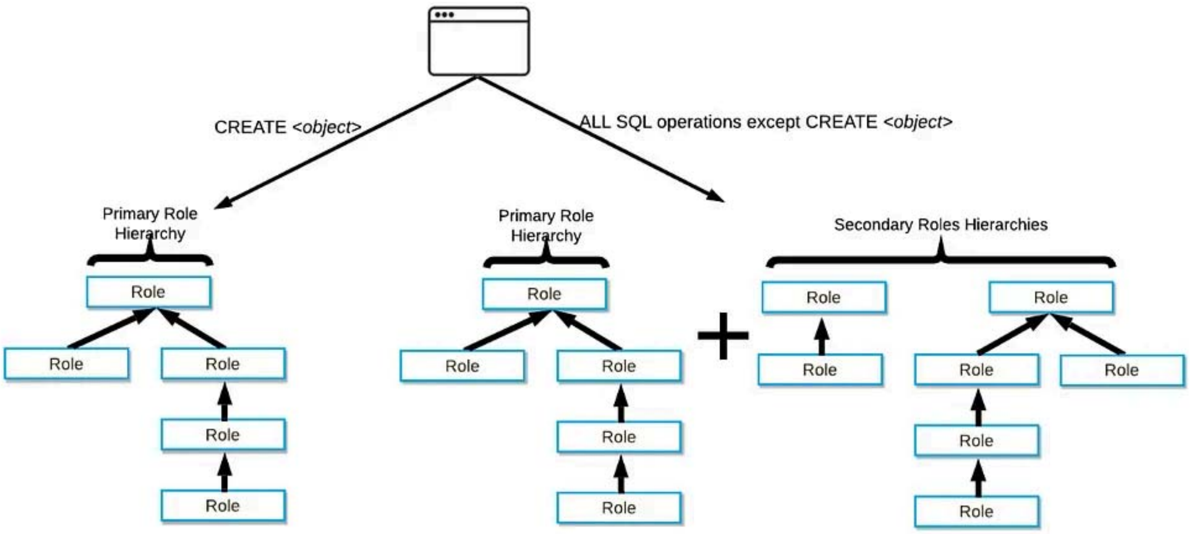
Every user/application who connects to snowflake will connect via some credential and role. **Primary role** is very well explained in snowflake documentation as below:

1. If a role was specified as part of the connection and that role is a role that has already been granted to the connecting user, the specified role becomes the current role.
2. If no role was specified and a default role has been set for the connecting user, that role becomes the current role.
3. If no role was specified and a default role has not been set for the connecting user, the system role PUBLIC is used.

Any other role which is granted to the user but not set as primary role is **secondary role.** In short, every user has multiple role access but at a time only 1 role can become primary and rest all other roles are secondary roles. A user can activate their secondary role in a session by using "USE SECONDARY ROLE" in snowflake. Then you might think what is the difference between both the role both are same? Getting confused ? Let's decode this now.



To create any object you have to use primary role authorization in snowflake and later ownership of that object is assigned to primary role. Secondary role can be used to do any permitted action except creation of object. A very nice diagram from Snowflakes documentation which says it all.



Source: Snowflake

. . .

Things to remember:

1. Access control consideration can be found **here**.
2. All kind of possible privilege in snowflake can be found **here**.
3. All basic example and code sample how RBAC works in snowflake can be found **here**.

. . .

Hope this blog helps you to get insight into the **Snowflake Access Control** . Feel free to ask a question in the comment section if you have any doubts regarding this. Give a clap if you like the blog. Stay connected to see many more such cool stuff. Thanks for your support.

You Can Find Me:

Subscribe to my YouTube Channel: https://www.youtube.com/c/RajivGuptaEverydayLearning

Follow me on Medium: https://rajivgupta780184.medium.com/

Follow me on Twitter: https://twitter.com/RAJIVGUPTA780

Connect with me on LinkedIn: https://www.linkedin.com/in/rajiv-gupta-618b0228/

#Keep learning #Keep Sharing #RajivGuptaEverydayLearning #SnowflakeDataSuperhero #RajivGupta

Snowflake Rajiv Gupta Data Superhero Rbac Security

Published in Snowflake Builders Blog: Data Engineers, App Developers, AI, & Data Science

Following

10K followers · Last published 15 hours ago

Best practices, tips & tricks from Snowflake experts and community

Written by Rajiv Gupta

Follow

1.6K followers · 18 following

Snowflake Data Super Hero, Director Of Technology at Kipi.ai , Snowflake SME

Responses (1)

Eng Omar Essam

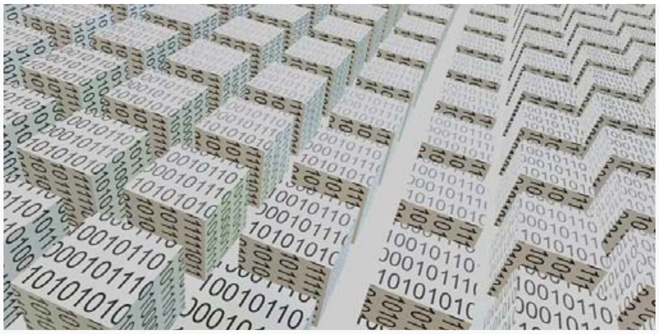
What are your thoughts?

M Manmohan Shah
Dec 24, 2022

Thanks, now its really easy to understand. Appreciate to co-relate with easy terminology to make it more easy.

Reply

More from Rajiv Gupta and Snowflake Builders Blog: Data Engineers, App Developers, AI, & Data Science

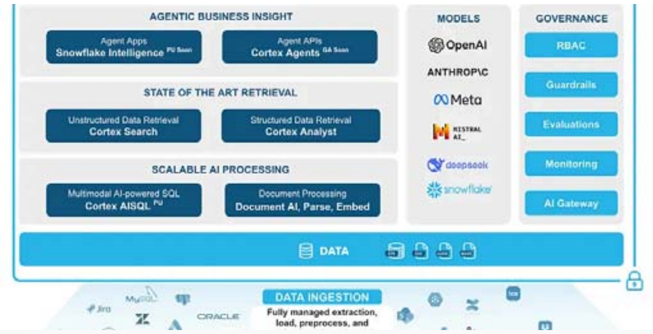


In FAUN.dev() 🐾 by Rajiv Gupta

Automating Data Governance with Snowflake's Sensitive Data…

In the age of data democratization, balancing accessibility with privacy is no longer option…

Sep 29 👏 5


AGENTIC BUSINESS INSIGHT
Agent Apps Snowflake Intelligence
Agent APIs Cortex Agents
STATE OF THE ART RETRIEVAL
Unstructured Data Retrieval Cortex Search
Structured Data Retrieval Cortex Analyst
SCALABLE AI PROCESSING
Multimodal AI-powered SQL Cortex AISQL
Document Processing Document AI, Parse, Embed
DATA
MODELS
OpenAI
ANTHROPIC
Meta
MISTRAL AI
deepseek
snowflake
GOVERNANCE
RBAC
Guardrails
Evaluations
Monitoring
AI Gateway
DATA INGESTION
Fully managed extraction, load, preprocess, and
ORACLE

In Snowflake Builders Blog: Data Engine… by Ume…

Build Snowflake Cost Savings and Performance Agent in 5 minutes

Snowflake Cortex Agent, building agentic AI solutions is no longer about complexity—it's…

Sep 30 👏 44 💬 3

In Snowflake Builders Blog: Data Enginee… by Sai…

Agent Instruction Best Practices for Snowflake Intelligence

Prototyping AI agents is easy. However, successfully launching reliable agents to…

Sep 25 👏 54 💬 3

In Snowflake Builders Blog: Data Engine… by Raji…

Using Contacts in Snowflake: A Strategic Guide to Schema-Level…

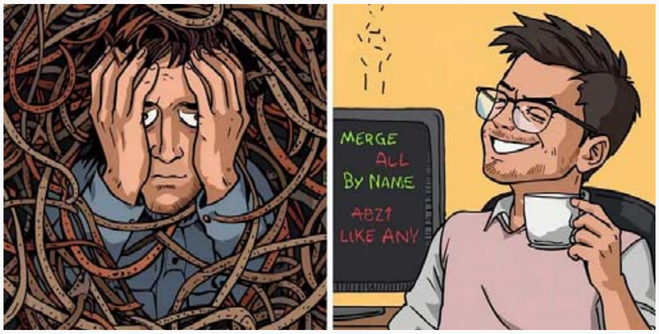
In modern data ecosystems, clarity in ownership and support is critical. Snowflake'…

Oct 10 👏 2

See all from Rajiv Gupta

See all from Snowflake Builders Blog: Data Engineers, App Developers, AI, & Data Science

Recommended from Medium



Vishal Kaushal

Top 10 Snowflake SQL Functions Every Data Professional Should…

Are you spending too much time writing and maintaining long SQL queries in Snowflake?…

✦ Oct 16 👏 34 💬 1

In Long. Sweet. Valuable. by Ossai Chinedum

I'll Instantly Know You Used Chat Gpt If I See This

Trust me you're not as slick as you think

✦ May 16 👏 25K 💬 1491

In Coding Nexus by Civil Learning

I Handed ChatGPT $100 to Trade Stocks — Here's What Happened i…

What happens when you let a chatbot play Wall Street? It's up 29% while the S&P 500…

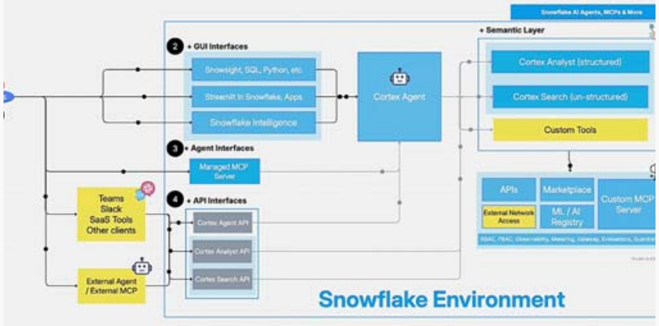
John Ryan

Snowflake Gen-2 Warehouses: Faster Performance or Just Highe…

Introduction

Sep 3 6.1K 162

Oct 1 6



In Fru.dev by Fru

5 Strategic Pillars to Understanding Snowflake's AI…

A Complete Architecture Overview

Sep 16 23

In Write A Catalyst by Utsuk Agarwal

The 10 Morning Habits That Quietly Make You Unstoppable

I Tried Them for 6 Months — The Results Were Unreal

Aug 11 14.1K 475

See more recommendations