Continuus TECHNOLOGIES

CASE STUDY

Automating Snowflake and AWS Integration with Terraform

By using Terraform to automate the manual steps in integrating Snowflake with AWS, our team streamlined the setup process, reduced human error, and saved significant time and resources for clients.

AT A GLANCE

ABOUT THE CLIENT

Data Management department at an Asset management, capital market, and investment management organization

PROBLEM

Setting up Snowflake pipelines with AWS integration required 30-40 manual steps, making the process time-consuming and prone to errors

SOLUTION

Terraform to automate the Snowflake/AWS integration process, creating reusable code that streamlined the setup and eliminated the need for manual intervention

OUTCOME

Saved significant time and resources for clients, minimized human error, and ensured a more efficient and maintainable integration process

FRAMEWORK

- DevOps and CI/CD, Configuration Management
- Snowflake, Terraform, Hashicorp, AWS, GitLab, Flyway by Redgate

PROBLEM

Our team was initially brough in to assist with setting up Snowflake pipelines using GitLab. We set up the code and infrastructure and code in Terraform in order to automate many parts of that process. The pipeline set up required some aspects of Snowflake to integrate with a cloud provider, in this case Amazon Web Services (AWS). When setting up the Snowflake/AWS integration, there are typically 30-40 manual steps to complete. We used Terraform to write a code in a universal language and streamline, automate, and speed up the Snowflake/AWS integration.

SOLUTION

Terraform is the key to setting up the necessary building blocks for the infrastructure and tool stack. It uses the universal Hashicorp language that acts as a wrapper around everything to make it easily consumable and usable for other users to understand and use – Azure pro, AWS wiz, or something else. We used Flyway by Redgate for SQL queries and auxiliary pipelines for SQL changes.

With Terraform, we wrote code that automatically passes the necessary outputs back and forth to each other to stand up the Snowflake/AWS integration and bypassed the approximately 40 manual steps that otherwise needed to be completed.

Once the code has been defined, any environment can be set up using that code set in production; the only thing that changes is the variables passed.

Following an agile DevOps CI/CD framework and “fast fail” mentality, we received instant feedback while standing up the code and testing the pipelines and were able to quickly resolve any issues that arose. Ongoing support and troubleshooting from our team enabled the clients’ teams to continue their work with minimal delays.

OUTCOME

We created code using Terraform to automate the tedious manual Snowflake/AWS integration process, using a universal language to make it accessible to a wider group of users and cutting out the margin for human error.

Many clients are happy with Snowflake once they are all set up in it, but the automation of that set up is what is missing and they don’t want to deal with the manual process provided. That process is time-consuming and can be difficult to maintain down the line. Using an agile DevOps framework, we were able to test at every step of the way and avoid issues later on in production. Overall, automating this process saves time and resources for the client.