Introducing the Snowflake Handler in Oracle GoldenGate for Big Data

Edward Whalen
Chief Technologist
Performance Tuning Corporation
www.perftuning.com

Oracle has just introduced a new version of Oracle GoldenGate for Big Data that includes support for several new platforms, including the Stage and Merge feature for Snowflake. See https://blogs.oracle.com/dataintegration/post/goldengate-for-big-data-214000-is-now-available.
It provides much-improved support for replicating to Snowflake than what was available in the past.

With the expanded support for Snowflake, GoldenGate customers can now replicate data from any supported data source (Microsoft SQL Server, PostgreSQL, Oracle, Kafka, MySQL, MongoDB and more) into Snowflake. It opens up significant opportunities to derive value out of data residing on-premises or in the Cloud using GoldenGate and Snowflake together.

Prior to the Stage and Merge feature there were three options available to replicate data into Snowflake using GoldenGate for Big Data:

- replicating directly to Snowflake using JDBC
- replicating into Apache Kafka and then connecting Snowflake to Kafka manually
- replicating to an Object Storage technology such as Amazon Web Services (AWS) S3 Buckets or Azure Data Lake Storage and using an ETL tool or a Snowflake utility to load the database.

The first option is not as optimal from a performance standpoint and requires writing your own SQL statements. The second option will scale and requires some manual code that comes with its own set of issues. The third option is technically the most efficient, but it requires an ETL tool or some utilities. With this new release of GoldenGate for Big Data, Oracle took the best technical approach and automated it to simplify the user experience and reduce the time it takes to load real-time data into Snowflake.

What is Stage and Merge?

GoldenGate for Big Data provides an optimized approach for loading data into data warehouse targets such as Snowflake that support Massively Parallel Processing (MPP). In such environment, it is more efficient to load the target using a micro-batching approach than to replicate every single Data Manipulation Language (DML) operation.

The change data coming from the trail files is first going through an Operation Aggregation phase which is unique to Oracle. GoldenGate merges multiple operations that happened on the same row into a single output operation to reduce the amount of data sent to the target, optimize performance and lower costs. Once the data has been compressed/aggregated, it is staged in micro-batches to a staging location and then merged into the data warehouse target tables using merge SQL statements.

The Stage and Merge approach is already available for many platforms, including the Oracle Autonomous Data Warehouse or Azure Synapse Analytics. We can now benefit from it to load into Snowflake!

Stage and Merge for Snowflake

Stage and Merge for Snowflake can leverage two types of stages for storing data: internal stage and external stage. With internal stage, Oracle GoldenGate will automatically create a staging area within Snowflake to receive the data, which in turn is merged to the Snowflake database. With external stage, users must first create a bucket and an external stage integration in Snowflake with Amazon S3, Azure Data Lake Storage or Google Cloud Storage. This external stage location is then used as the staging area, GoldenGate will then create an external table in Snowflake to read the staged data and merge it into the final target tables.

Configuring GoldenGate Replicat for Snowflake

The Snowflake Stage and Merge option for GoldenGate for Big Data 21.4 requires JDK 8. JDK 8 can be downloaded from https://www.oracle.com/java/technologies/downloads. It is necessary for the handler to run correctly.

I recommend running the GoldenGate for Big Data Microservices version, and this paper will explain how to use it. For testing purposes, I have also installed GoldenGate for Big Data 21.4 Microservices.

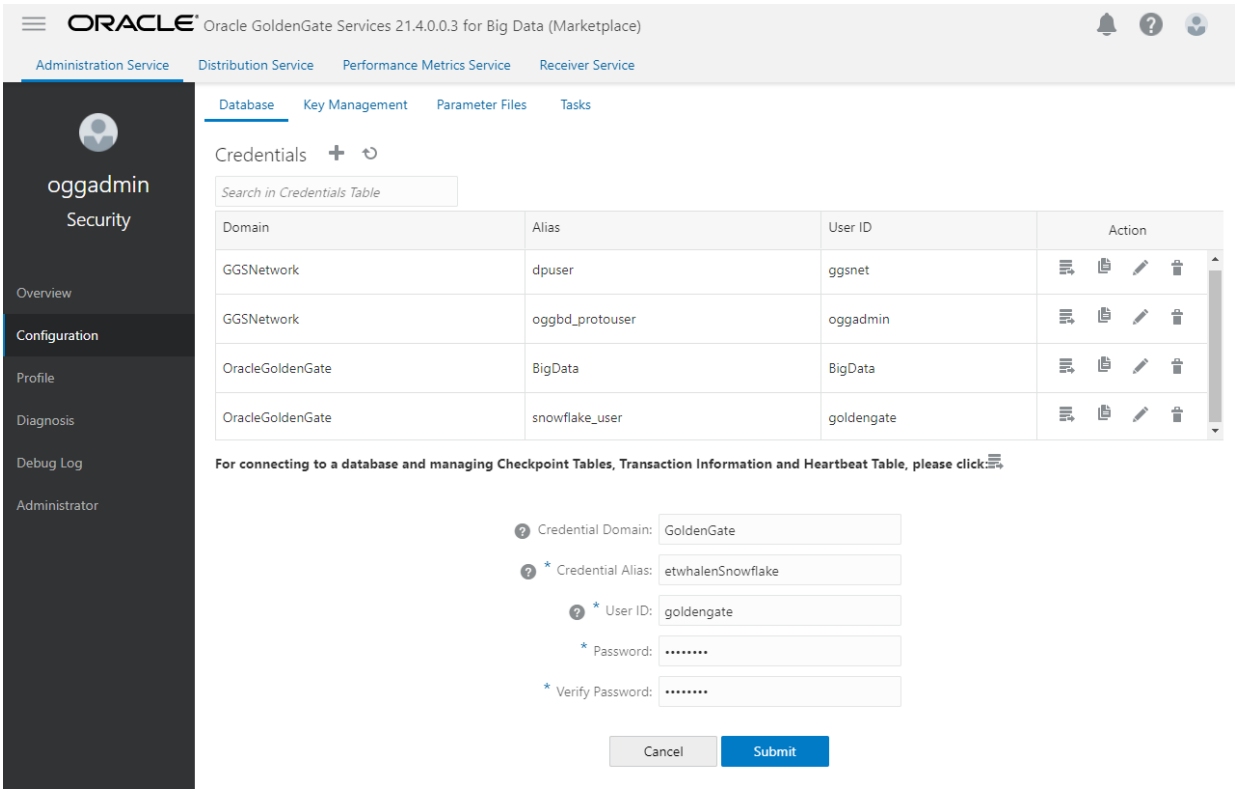
For those of you who are not familiar with it, Microservices is an entirely different way of managing Oracle GoldenGate. The extract and replicat are similar but how they are managed and how trail files are moved between source and target systems is different.

If you aren't familiar with GoldenGate Microservices for Big Data, you must first install GoldenGate and then create a deployment. A deployment is an instantiation of the software and includes its own services. You can have multiple deployments from the same software release.

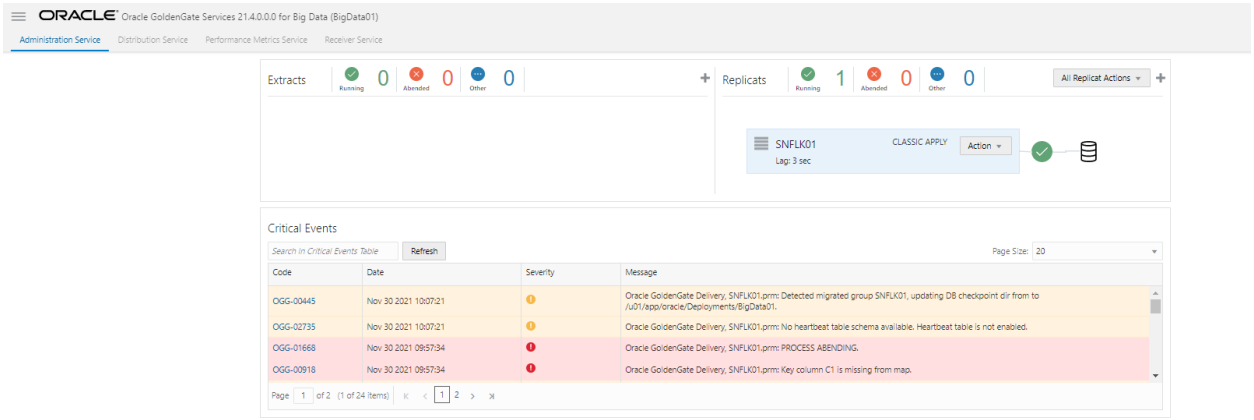
Prior to creating a replicat download the dependencies needed by that replicat. The latest versions of GoldenGate for Big Data now come with a set of scripts to download required dependencies automatically. You will find several scripts for various Big Data target types in the $OGG_HOME/opt/DependencyDownloader directory. For this use case, run snowflake.sh. This will download the latest version of the Snowflake JDBC driver into the $OGG_HOME/opt/DependencyDownloader/dependencies directory. This will be important later.

Once the Big Data deployment is created and running, a replicat can be created from the GoldenGate Big Data GUI. The is done from the Administration Service.

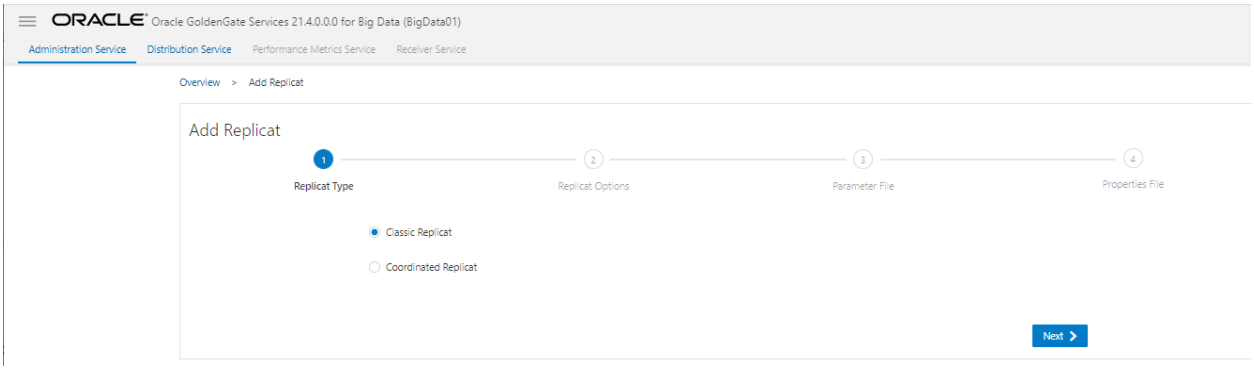
Before we create a Replicat, we will store the Snowflake credentials in the GoldenGate credential store. To do so, click on the Administration Server menu, click Configuration and add a Credential using the + icon. Enter a Credential Domain, a Credential Alias, and your Snowflake user as User ID and Snowflake password as Password.



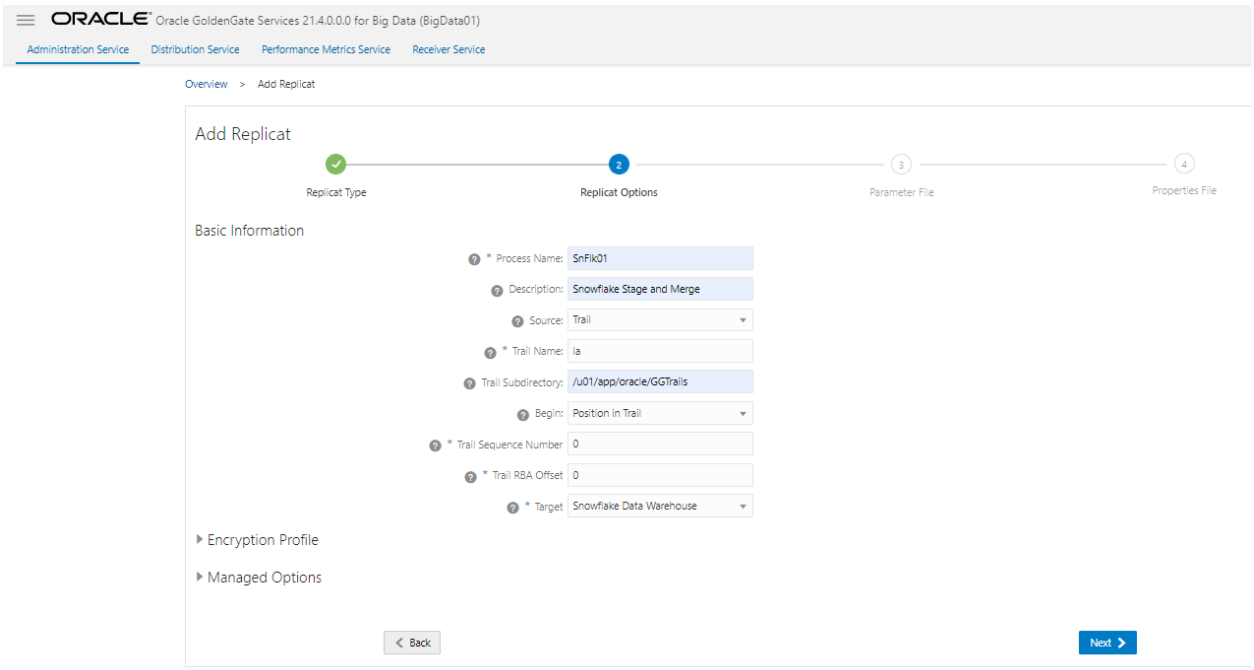
Next, from the Administration Service, click the + icon next to Replicats



From here, select Classic or Coordinated Replicat. For this example, I am choosing Classic. Coordinated replicat allows for some parallelism in the replicat by setting up multiple threads. It can provide for better performance but does require some additional setup and tuning.



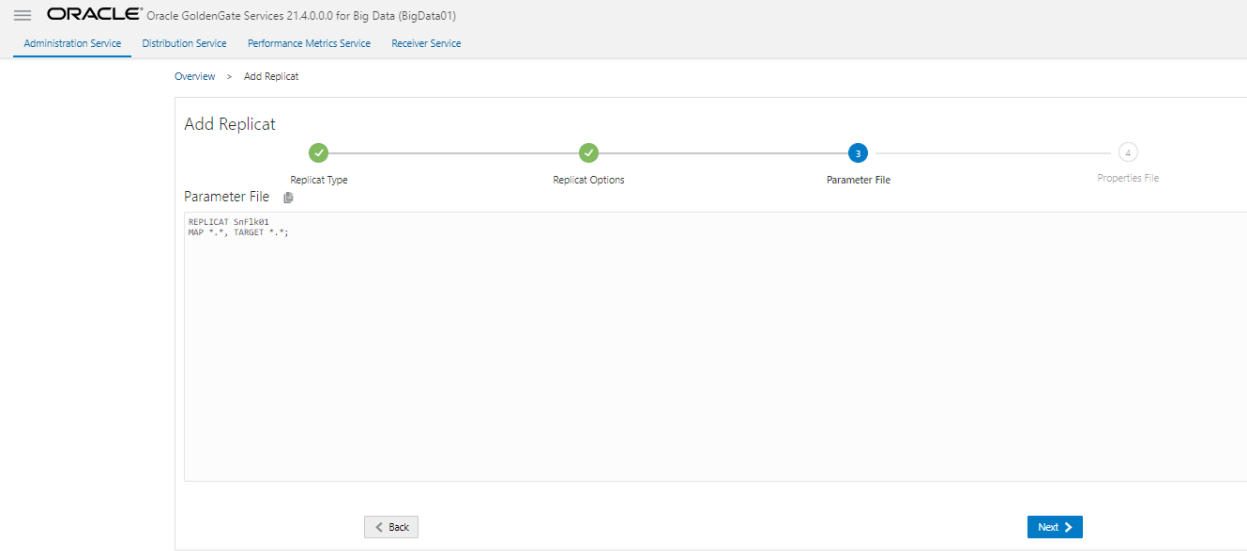
From the next menu fill in the replicat name, trail and from the Target menu begin typing Snowflake until the Snowflake Data Warehouse option appears and choose it. This is important as, by default, you won't see Snowflake in the list.



If this replicat is using data from another GoldenGate instance via the Distribution and Receiver Services, you must make sure to setup the source and trail name properly. By default, the trail file directory is under the Deployment home in the ./var/lib/data subdirectory.

Continue filling out the screen and click next

The default parameter file should be sufficient.



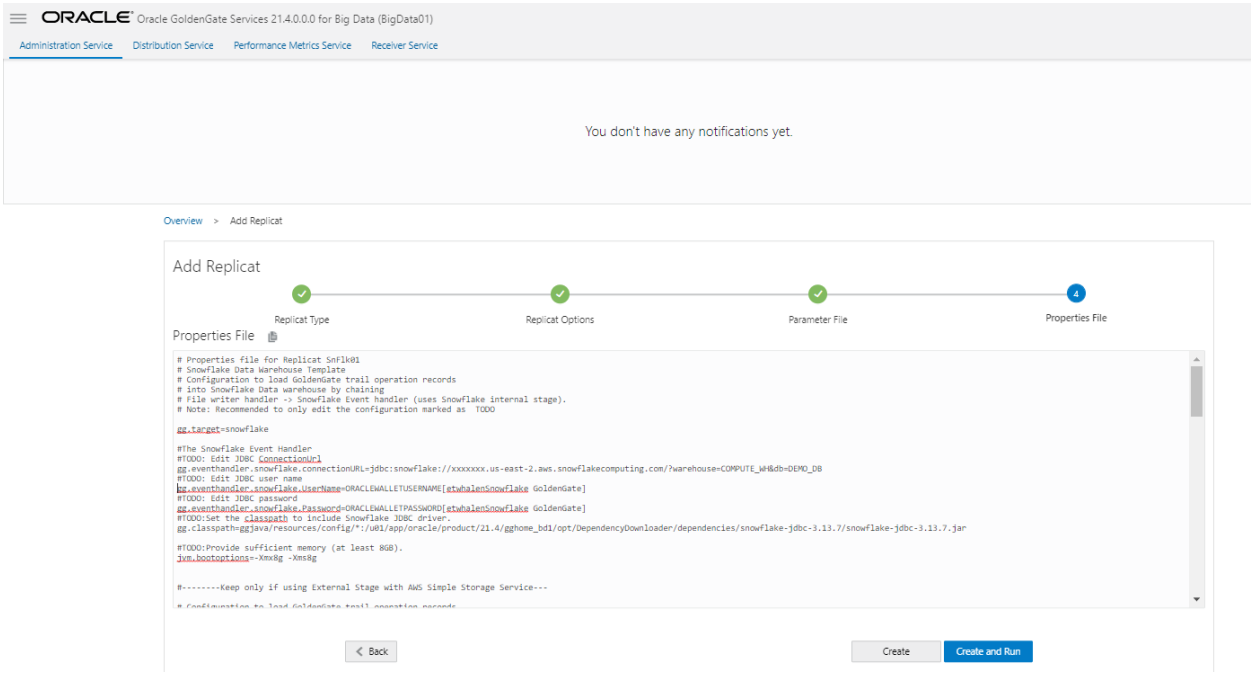
The next screen is where you define the properties required by the handler to load into Snowflake. If you are using internal stage, you only need to configure the section at the top with the following properties:

- gg.eventhandler.snowflake.connectionURL: This represents the JDBC URL that GoldenGate will use to connect to Snowflake and run some commands.
  Note: Snowflake automatically disconnects the JDBC connection after some time. You can add the `&CLIENT_SESSION_KEEP_ALIVE=true` parameter to the JDBC URL to overcome this problem
- gg.eventhandler.snowflake.UserName: This property stores your Snowflake username. It can be added to the GoldenGate credential store to avoid storing this value in the Properties file. If you do so, you must use the ORACLEWALLETUSERNAME[alias domain] function to retrieve the stored value. Review the documentation for more info about this function.
- gg.eventhandler.snowflake.Password: This property stores your Snowflake password. I recommend adding it to the GoldenGate credential store to avoid keeping this value in the Properties file. If you do so, you must use the ORACLEWALLETPASSWORD[alias domain] function to retrieve the stored value. Review the documentation for more info about this function.
- gg.aggregate.operations.flush.interval: By default, GoldenGate perform its stage and merge cycle every 30 seconds. It is possible to increase or decrease this value using this parameter, it is expressed in milliseconds. It is recommended to use the flush interval parameter with caution as the higher the value is the more memory the Replicat will require. This can lead to out of memory errors and potentially stop the machine the Replicat is running on. I am using the default value and didn't specify the parameter in the Properties file.

Notice that I'm using Wallet credentials that I created earlier. This is a best practice to not include actual usernames and passwords in the parameter files.

```
#The Snowflake Event Handler
#TODO: Edit JDBC ConnectionUrl
```

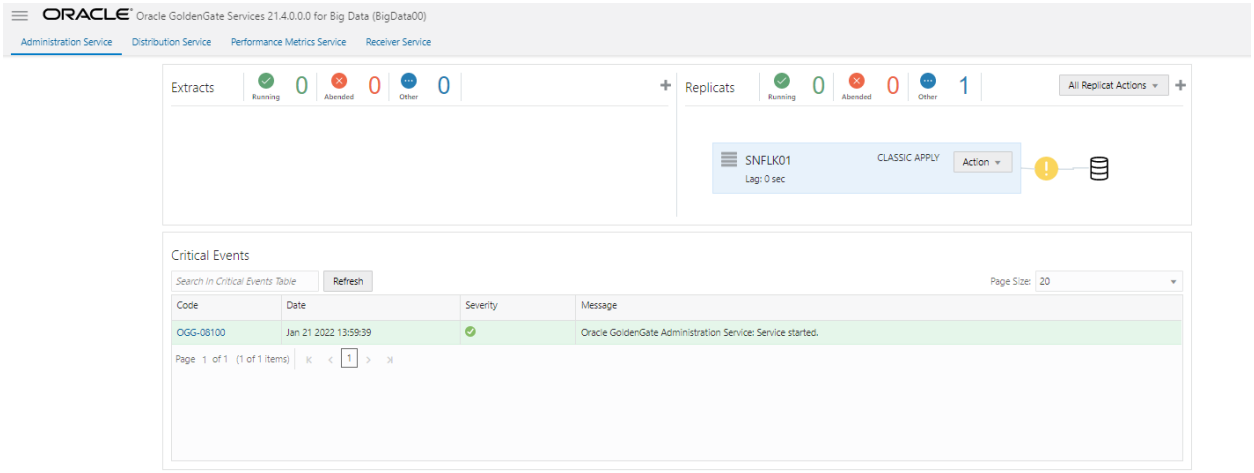
```
gg.eventhandler.snowflake.connectionURL=jdbc:snowflake://xxxxxxx.us-east-
2.aws.snowflakecomputing.com/?warehouse=COMPUTE_WH&db=DEMO_DB
#TODO: Edit JDBC user name
gg.eventhandler.snowflake.UserName=ORACLEWALLETUSERNAME[etwhalenSnowflake
GoldenGate]
#TODO: Edit JDBC password
gg.eventhandler.snowflake.Password=ORACLEWALLETPASSWORD[etwhalenSnowflake
GoldenGate]
#TODO:Set the classpath to include Snowflake JDBC driver.
gg.classpath=ggjava/resources/config/*:/u01/app/oracle/product/21.4/gghome_bd1/opt/Depe
ndencyDownloader/dependencies/snowflake-jdbc-3.13.7/snowflake-jdbc-3.13.7.jar
#TODO:Provide sufficient memory (at least 8GB).
jvm.bootoptions=-Xmx8g -Xms8g
```



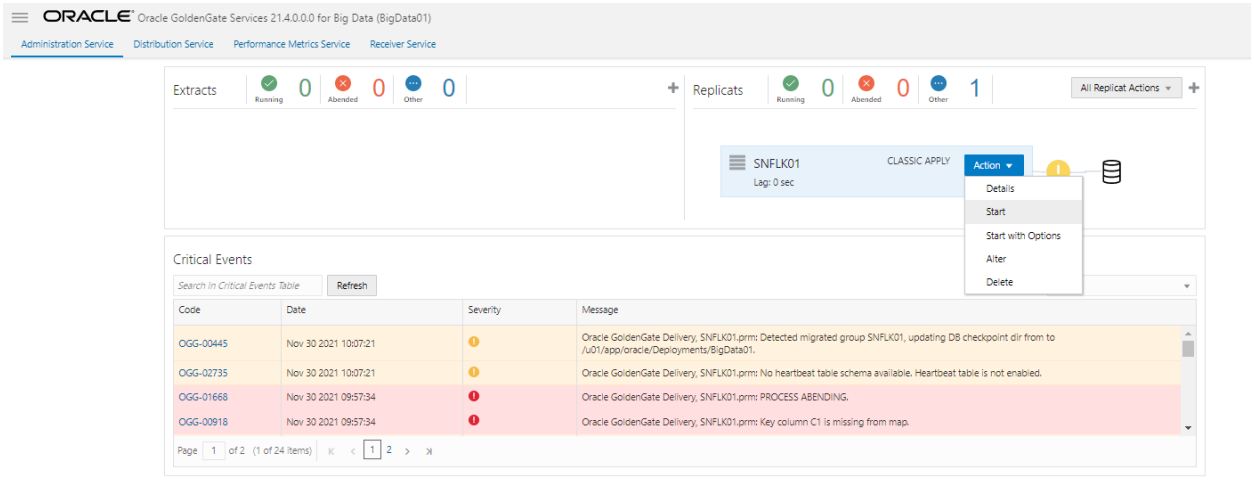
Suppose you decide to use external stage on Amazon Web Services (AWS), Microsoft Azure or Google Cloud Platform. In that case, you will first need to create the object storage bucket and configure the external stage integration in Snowflake. When this is done, locate the section of the Properties file applying to your cloud provider and remove the rest. Finally configure the appropriate properties, such as the AWS region (*gg.eventhandler.s3.region*) or the external stage integration (*gg.eventhandler.snowflake.storageIntegration*).

Finally, click Create and Run to create the Replicat and run it. Using the Snowflake web interface, you can make some changes in your source system and see the data replicated in real-time.

You can also click Create to create the process, double-check your settings, and finally start the Replicat from the Action dropdown.



Once you have started it will take a few seconds for the replicat to respond.



If everything is good, the replicat will start. Once it has started you will know that you are connected to Snowflake.

Additional Considerations

In this paper, I have installed Oracle GoldenGate for Big Data 21.4 Microservices on a Compute instance running on AWS, but other options are available. For example, GoldenGate for Big Data is available on the Oracle Cloud Infrastructure (OCI) Marketplace. This Marketplace offering automates the installation and configuration, making it simple to get started quickly. It is also possible to run GoldenGate on other cloud providers or on-premises.

It is recommended to co-locate or have GoldenGate for Big Data and Snowflake as close to each other as possible to avoid introducing any unnecessary network latency. This can be easily achieved given the modular and decentralized architecture of GoldenGate, making it a perfect fit for modern hybrid, multi-cloud environments.

Conclusions

Using the stage and merge feature of Oracle GoldenGate for Big Data 21.4 provides a high-performance, reliable method of replicating data into Snowflake from any supported source system (databases, NoSQL, Kafka, among others). It offers much better performance and ease of use than traditional methods of replicating data into Snowflake, such as using JDBC or third-party ETL tools unless those tools are needed for data transformation.

In addition, we have seen that GoldenGate can be installed anywhere: on-premises, on Oracle Cloud, or third-party clouds to be as close as possible to Snowflake and your other data sources. It is especially important for modern and distributed data architectures such as data mesh.

Oracle has made a significant effort to improve the ease of use and user experience in GoldenGate for Big Data 21.4 Microservices. You will notice the improvements as soon as you try it.

For more information or help with anything outlined in this article, feel free to contact us at support@perftuning.com