MICROSTRATEGY® WORKS WITH SNOWFLAKE

MicroStrategy Incorporated

TABLE OF CONTENTS

0 INTENT OF THIS DOCUMENT

The intent of this document is to give an overview of different technical aspects to consider when integrating MicroStrategy with Snowflake.

This document was created by thorough research and customer site visits to understand different customers' use cases of Snowflake with MicroStrategy. This document briefly summarizes common challenges with their resolutions and provides reference links for detailed information.

Note: MicroStrategy and Snowflake are both changing rapidly. If you encounter a situation where the document is not valid, please let us know and we will update it.

Author: Talhah Zafar (PM)
Collaborators: Marcin Graczyk (ES), Sergio Sainz (TEC), Yang Huang (TEC), Chris Van Huis (Snowflake)
Date of Original Creation: 2021Q1
Last Update: 2021Q1

1 OVERVIEW

1.1 MicroStrategy

MicroStrategy is the leading enterprise business intelligence tool with its mission statement to make every enterprise a more intelligent enterprise.

Learn more at MicroStrategy.

1.2 Snowflake

Conventional data warehouses weren't designed to keep up with the exploding demand for data-driven insight at modern organizations. With years of experience building the world most performant databases, Snowflake founders were in a unique position to see this problem, and how cloud could solve it. In 2012, they founded Snowflake with the simple intention of enabling the worlds organizations to be data driven. With a brand-new architecture for a data platform designed to take advantage of cloud elasticity and simplicity, they created Snowflake.

A few key features:

- Very easy to use
- Near zero infrastructure
- Secure & highly available
- No indexes, distribution keys, partitioning, or vacuuming
- Data storage is columnar compressed and encrypted
- Only pay for what is used: Scale compute up and down and take advantage of multiple workloads simultaneously

Learn more at Snowflake.

1.3 MicroStrategy and Snowflake Better Together

You can learn more about MicroStrategy's Integration with Snowflake from "MicroStrategy Works with Snowflake" session.

2 RELEASE HIGHLIGHTS

2.1 MicroStrategy 2021

- 12% OOTB Performance Improvements with Implicit Table Type Creation.
- OIDC Support for Snowflake.
- Unified Quoting Updates to better handle characters.
- 30% Performance Improvement with Lookup Cube Caching.

2.2 MicroStrategy 2020

- Up to 3X Performance Improvements OOTB using Derived Table Syntax and Arrow Data Transfers.
- Connectivity Wizard Support for Snowflake
- ODBC & JDBC drivers shipping OOTB.
- Ship latest JDBC driver with Arrow Format enabled.
- OAuth Support of Snowflake with Okta and Azure AD.
- SSO Support of Snowflake with Azure AD.

3 TECHNICAL CONSIDERATIONS WITH SNOWFLAKE

3.1 Feature support

When migrating existing applications to Snowflake, certain MicroStrategy capabilities are leveraged with tight integration of the databases. It is important to understand that certain capabilities powering the user workflows might not be supported in Snowflake.

Below are the common database workflows MicroStrategy customers face when dealing with Snowflake:

Feature	Snowflake	MicroStrategy
SSO	Yes	Yes
OAuth 2.0	Yes	Yes
Kerberos	No	Yes
SAML	Yes	Yes
Parameterized Queries	Yes	Yes
Data Import	Yes	Yes
Live Connect	Yes	Yes
Arrow Format	Yes	Yes
Materialized Views	Yes	Yes
Search Optimization	Yes	Yes
Binary Data Type	Yes	Yes

3.1.1 SSO

Single Sign-On (SSO) allows for one-time login for both MicroStrategy and Snowflake.

Impact: Authentication

Motivation: One-time login

Details: MicroStrategy currently support SSO with Azure AD. After the Administrator sets up SSO, they only have to login once and take advantage of both MicroStrategy and Snowflake from the same credentials.

To learn more about this functionality, please take advantage of the Integrating MicroStrategy with Snowflake for Single Sign-On using Azure AD documentation.

3.1.2 OAuth 2.0

OAuth 2.0 is becoming very widely used these days and is one of the most common authentication methods out there.

Impact: Authentication

Motivation: Use non-native authentication

Details: MicroStrategy supports both Azure AD and Okta Authentication with Snowflake. Once the administrator sets it up, the end user has to just log in with their Okta or Azure AD email credentials to start using MicroStrategy with Snowflake.

MicroStrategy OAuth 2.0 workflow with Okta can be seen here. Please follow this product guide to implement OAuth 2.0 with MicroStrategy.

3.1.3 Kerberos

Kerberos helps with authentication.

Impact: Authentication

Motivation: Customers want to leverage Kerberos authentication for their infrastructure

Details: Snowflake doesn't support Kerberos and the Snowflake customers are directed to use SAML authentication instead.

3.1.4 SAML

SAML helps with authentication.

Impact: Authentication

Motivation: Customers want to leverage SAML authentication for their infrastructure

Details: Snowflake supports SAML authentication. Please refer to this documentation to take advantage of SAML authentication .

3.1.5 Parameterized Queries

MicroStrategy's support for parameterized queries can improve performance in scenarios that require the insertion of information into a database.

Impact: Performance, security

Motivation: Customers can write data into tables faster

Details: Parameterized queries are SQL queries that can use placeholders for data. Using placeholders allows these queries to be re-used. A common application of this re-usability is to combine multiple inserts of data into a database as a single query. The following is an example of a parameterized query: INSERT INTO DMTABLE (Customer_ID, Customer_Name) VALUES (?, ?) Combining multiple INSERT statements into a single query can improve the performance of inserting data into the database.

Starting in MicroStrategy 2021 Update 1, parameterized queries are turned on by default.

3.1.6 Data Import

MicroStrategy supports data import with Snowflake.

Impact: Capability

Motivation: Customers can bring in data as a cache so they can save on costs using Snowflake

Details: With the Data Import functionality, MicroStrategy imports data from Snowflake into its fast in-memory cubes and can constantly send queries to the data in the memory saving on compute costs. Unlike most traditional warehouses, result fetching is optimized in Snowflake for cubes, so no additional tooling is required for bulk imports. Although Snowflake caches query results for up to 24 hours on their side, MicroStrategy has the capability of storing the data for much longer depending on the refresh interval.

Example: Updating cache once a month, using aggregated data, and saving on the cost by not querying data frequently in Snowflake. Use a cube with MicroStrategy if you're only concerned with last month's aggregated data. The cube can answer most daily questions instead of going back to Snowflake.

3.1.7 Live Connect

MicroStrategy support live-connect with Snowflake.

Impact: Capability

Motivation: Customers can create ad-hoc visualizations

Details: With live connect functionality, customers can create ad-hoc visualization with Snowflake without importing the data back into MicroStrategy. Most customers take advantage of quick response time offered by dossiers or reports by querying a direct and concise slice of data utilizing the power of Snowflake.

Example: Use live connect if you're trying to look at certain transaction details, such as transactions between 11:20AM UTC – 11:25AM UTC that originated from Virginia, and the product category Electronics.

3.1.8 Arrow Format

Arrow Format helps with performance

Impact: Performance

Motivation: Customers can access data faster

Details: Arrow format is an internal Snowflake performance improvement and doesn't affect customer workflows. At the time of publishing this document, only the current JDBC driver supports the Arrow format functionality. ODBC driver doesn't support arrow format, but Snowflake plans to roll out an update to support it in the near future. MicroStrategy 2021 can take advantage of Arrow Format provided by Snowflake which can help improve the performance for customers. Description and benchmarks are available here.

3.2 Performance

Performance is an important consideration with MicroStrategy and Snowflake. MicroStrategy creates optimized SQL for Snowflake and it is always recommended to be on the latest MicroStrategy release to take advantage of the most optimal performance. Below are some performance considerations which will help you achieve better performance with MicroStrategy.

3.2.1 Using VARCHAR Size When Creating Tables

Queries can have a significant increase in the fetch time when there are tables containing VARCHAR without specifying a size in Snowflake. Performance degradation occurs because the MicroStrategy Intelligence Server uses an ODBC call (SQLDescribeCol) to determine the proper resources to allocate for the query results of VARCHAR columns. In anticipation of resources, a large memory will be allocated for a small amount of data (even if the string is only a few hundred characters maximum). VARCHAR (16777216) will set the limit for more size and will affect performance while fetching. Even if the tables are created, you can alter the VARCHAR size. Please refer to KB441548 to learn more about it.

3.2.2 Query Performance Comparison of Old DBMS vs Snowflake

Always use a similar hardware in Snowflake to your old DBMS solution if you're doing performance comparisons. For best performance analysis, make sure you're performing apple-to-apple comparisons when comparing your old DBMS to Snowflake.

For example: Queries tend to run faster in a powerful on-premise warehouse compared to a small instance of Snowflake. To match performance, please increase the instance size of Snowflake.

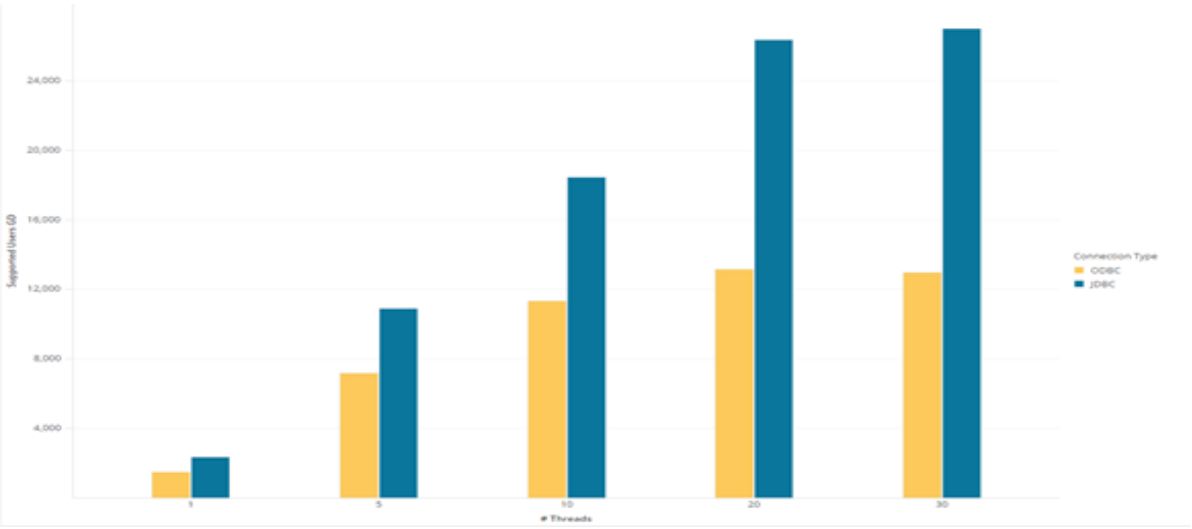
Please refer to this Snowflake blog to learn about comparing on-premise to cloud data platform. Learn more about different Snowflake warehouse considerations on the Snowflake documentation page.

3.2.3 Driver Optimization

3.2.3.1 ODBC vs JDBC

In MicroStrategy MicroStrategy 2021, both ODBC and JDBC are shipped out of the box.

Based on our internal benchmarks, JDBC connection shows better performance than ODBC connection. See the test result in the following (Figure 1).



One major reason is the adoption of Apache Arrow with the release of Snowflake JDBC drivers. You can read more about Snowflake's usage of Apache Arrow here.

Fetching result sets JDBC driver now leverages the Arrow columnar format to avoid the overhead previously associated with serializing and deserializing Snowflake data structures which are also in columnar format.

As of early 2021 Snowflake is in the process of adding Arrow support to the ODBC drivers. Contact Snowflake for more details.

3.2.3.2 Fetch Size Optimization

MicroStrategy only recommends this setting for cases when customers are having performance issues during ingestion of MicroStrategy cubes. MicroStrategy performed some internal testing on the fetch size and published KB484894.

You can enable this setting by modifying the connection string used in *Intelligence Server Database Connections* tab under the *Additional parameters* field. The following is an example connection string:

*JDBC;DRIVER={net.snowflake.client.jdbc.SnowflakeDriver};URL={jdbc:snowflake://sit.west-europe.azure.snowflakecomputing.com/?warehouse=YOUR_WAREHOUSE&useProxy=true&proxyHost=YOUR_HOST&proxyPort=8080};**FETCHSIZE=1000**;*

3.2.3.3 ODBC API Calls for Warehouse Catalog

MicroStrategy also supports using API calls to retrieve catalog information. However, due to the slow performance of Snowflake driver's metadata operations, this option is generally not recommended. The Snowflake API call we used, for example, get *Tables (null, null, …)*, by default returns all tables in all databases and schemas. Luckily, there is a setting on the Snowflake server side: CLIENT_METADATA_REQUEST_USE_CONNECTION_CTX. Read more about it here.

Setting this parameter to true would narrow the search scope to the current database and schema by the connection context.

3.2.4 Connection Caching

In MicroStrategy, there is a setting to keep the cache of the database connection. With this option enabled, MicroStrategy will cache the connection for the Snowflake warehouse. This will in return save couple of seconds every time a report is run as MicroStrategy and Snowflake doesn't have to create the connection again. This option is set by default for MicroStrategy customers.

Snowflake on the other hand also has a setting to keep the client connection alive. It is recommended to have it enabled to save on the time for connection creation.

To learn more about keeping your connection active, please read the FAQs for the ODBC/JDBC drivers. To learn more about the session parameters, please refer to the Snowflake documentation.

3.2.5 VLDB Properties

MicroStrategy constantly optimizes the queries and sets new defaults for the VLDB properties so customers can take advantage of the best performance possible. It is recommended to always stay on the latest MicroStrategy release so you can always have the best Snowflake experience OOTB.

Based on our customer insights, we have seen some performance improvements after tweaking some VLDB settings. There is no such thing as "One Size fits all" recommendation so please modify these settings responsibly. With the below VLDB settings, some customers achieved a 22%-36% performance improvements on pure snowflake SQL Execution. However, users should be aware of the fact that every implementation is different, and we give an idea where to look to increase performance.

- Join
  Base Table Join for Template = Use Fact Table
- Join
  From Clause Order = Move MQ Table in normal FROM clause order to the last (for RedBrick)
  Full Outer Join Support = Outer-Join

- Pre/Post Statements
  Drop Database Connection=Do not drop database connection after running user defined SQL [when using pre/post SQL]
- Query Optimization
  Sub Query Type = WHERE COL1 IN (SELECT s1.COL1...) falling back to EXISTS (SELECT col1, col2 ...) for multiple columns IN
  Transformation Formula Optimization = Always join with transformation table to perform transformation
- Select/Insert
  Distinct/Group by option = Use GROUP BY
  GROUP BY Non-ID Attribute = Use Group By
- Tables
  Fallback Table Type = Use Permanent table
  Intermediate Table Type=Derived Tables
  Table Creation Type = Implicit

3.2.6 Lookup Cube Caching

Every customer can take advantage of the lookup caching cube to improve report and cube performance by up to 30% against the data warehouse. Please refer to Improving Report and Cube Performance to learn more about it.

3.2.7 Snowflake Specific Tuning

Snowflake provides a query profile page which you can use to investigate your queries. For general tuning of Snowflake, please take advantage of the snowflake tuning page.

3.3 Connectivity

Connectivity is typically the biggest concern among customers. To better assist MicroStrategy customers, MicroStrategy has created documentation with the latest information.

3.3.1 MicroStrategy Official Documentation

View the MicroStrategy documentation, How to Connect to Snowflake with MicroStrategy.

The following Snowflake documents provide other relevant configuration and usage information that pertain to each driver:

- JDBC Driver
- ODBC Driver

3.3.2 Available Drivers

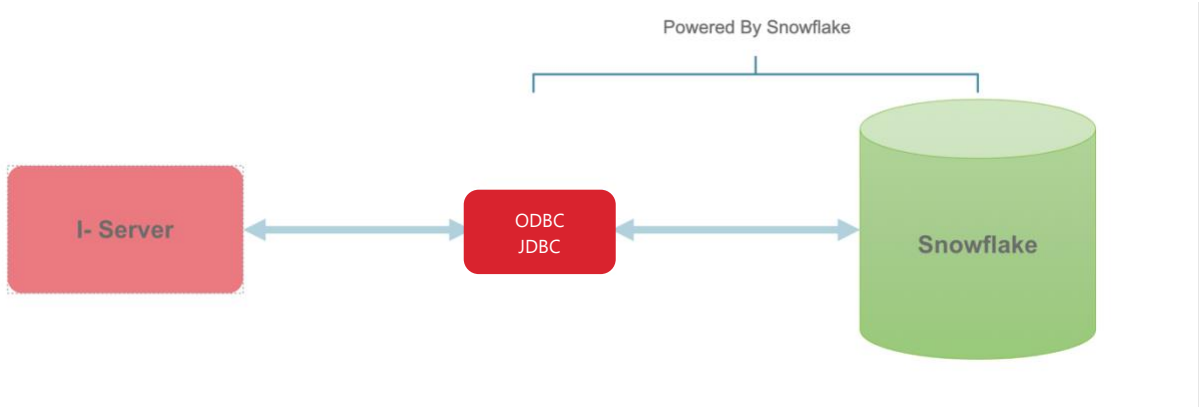
Starting in MicroStrategy 2020 Update 2, the Snowflake ODBC driver is shipped to simplify the connectivity workflow. In MicroStrategy 2021, MicroStrategy ships Snowflake JDBC out of the box. JDBC is more performant than the ODBC.

Always be aware of the latest driver version using the Snowflake release notes.

3.3.3 Which Driver is Best Suited?

MicroStrategy recommends using the JBDC driver for Snowflake because MicroStrategy customers have constantly seen JDBC driver outperform the ODBC driver. The Snowflake JDBC driver is built in house by Snowflake and tends to get the latest updates before the ODBC counterpart, developed by Simba.

3.3.4 Data Flow Diagram



4 COMMON ERRORS DEBUGGING WITH MICROSTRATEGY AND SNOWFLAKE

4.1 Debugging MicroStrategy and Snowflake

4.1.1 Unix Charset Issue

This usually arises when installing drivers directly from the Snowflake website instead of the MicroStrategy-provided drivers.

Issue: MicroStrategy displays partial catalog names (e.g., database, schema, table)

Solution:
1. Click Database Instance > Modify the Database connection > Advanced tab.
2. Configure the Database Connection in MicroStrategy Administrator to UTF-8.

Note: If the customer is using MSTR Intelligence Server on Linux/Unix they should do the opposite of the recommendation from the above. That is, choose non-UTF8 for Unix.

4.1.2 Debugging Initial DB Connection Time and Connection Caching

When working with Snowflake (which might be like other Cloud databases) we found an additional delta time of up to 2 seconds to establish a JDBC database connection compared to on prem installations. This time is required for spawning the JDBC wrapper and driver thread, completing the Cloud OSCP Certification check, and establishing proxy connection.

- How to detect that time:

In MicroStrategy Developer, the report SQL View shows this time as "Total Other Processing time" in summary and as "Other Processing Time" in the first SQL Pass.

- How to avoid:

By default, MicroStrategy uses database connection caching. After a report is run, we keep the connection open (DB connection timeout values apply). If the same user requests another report, we reuse this connection, avoiding initial connection time.

- Caveat when using PreSQL:

In our customer case, MicroStrategy initially prevented connection caching as the customer used some PreSQL for use schema or alter sessions to comment the query. As we could alter a session in

such a way that it effects further reports, MicroStrategy drops a connection after a report, that used Pre or Post SQL, has finished.

- One VLDB setting for Connection Cache:

There is another VLDB Setting (check Advanced settings first) in the Pre/Post Statements section.

Drop Database Connection=Do not drop database connection after running user defined SQL [when using pre/post SQL]

With that we can cache the connection with improves performance for subsequent report executions that can reuse the connection.

4.2 Additional Questions

Please reach out to the MicroStrategy or Snowflake community site.

4.3 Enhancement Requests

Please log a case for MicroStrategy.