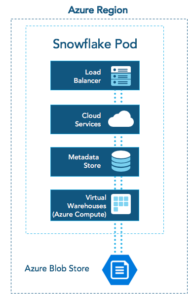
HOW TO GET STARTED WITH SNOWFLAKE ON AZURE

**The Technical Details**

Snowflake on Azure is architected to run on Azure, leveraging Azure compute and storage infrastructure services for data storage and query processing.

*Figure 1.*

As Figure 1 shows, Snowflake relies on Azure Blob Storage for data storage. To achieve scalable, highly performing data access, Snowflake stripes customer data across many storage accounts in Azure. Customer requests are processed by what we call virtual warehouses. A virtual warehouse is a set of virtual machines provisioned by Snowflake on Azure Compute. Each virtual warehouse with its virtual machines is dedicated to a single customer’s account. Snowflake receives requests via a load balancer. Requests leverage Snowflake’s cloud services and metadata layers for authentication, query compilation, transaction management, security, data sharing and other capabilities.

**First Steps on Azure**

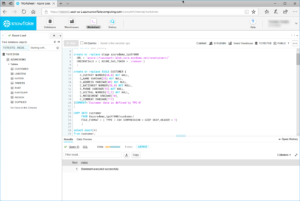
The most powerful insights often come from analytics that tie together different data sets. For this blog post, we will explore a scenario that uses Snowflake on Azure to correlate clickstream data from a customer-facing website with transactional data from an order processing system and visualize the results. To do that, we’ll use the following services in Azure:

* **Snowflake on Azure**: We’ll show you to connect to the Snowflake web UI to manage your Snowflake account, provision warehouses, explore your Snowflake databases, run queries, etc.
* **Azure Blob Storage**: In this example, Azure Blob Storage stages the load files from the order processing system.
* **Azure Data Lake Store**: The clickstream logs in this examples are stored in Azure Data Lake Store (Gen1) from where we will load them into Snowflake.
* **PowerBI**: Finally, we will connect PowerBI Desktop to Snowflake on Azure to visualize the results of the analytics.

The following paragraphs walk you through the different Azure data services and explain how to use them together with Snowflake on Azure.

## SNOWFLAKE ON AZURE: USING SNOWFLAKE THROUGH THE WEB UI

Customers access their Snowflake account on Azure using a URL such as https://<accountname>.east-us-2.azure.snowflakecomputing.com/. After authenticating, you can use the familiar Snowflake web UI to manage your databases, warehouses and worksheets, and access your query history and account details. The screenshot below (Figure 2) shows a Snowflake worksheet with the object explorer on the left, the query editor in the center and the query results at the bottom.



*Figure 2.*

In our example scenario, we’ll use the Snowflake web UI to submit the COPY statements to load data from Azure Blob Storage into Snowflake databases – which we will explain next.

**Azure Blob Storage: Loading data from Azure Blob Storage into Snowflake**

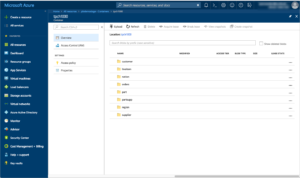
In this example, we assume that you already exported the data from the transactional order processing system into load files in Azure Blob Storage. Once the data is in Azure Blob Storage, you can load it into Snowflake using the existing support for external stages in Azure Storage ([https://www.snowflake.com/global-snowflake-loading-data-into-snowflake-from-azure-blob-storage/](https://www.snowflake.com/blog/global-snowflake-loading-data-into-snowflake-from-azure-blob-storage/)). Now, you can use these familiar steps to create a stage in Azure storage and run the COPY command to load the data:

1. Store your load files in an Azure Blob Storage container.
2. Create a Snowflake stage object referring to the blob storage location in its URL parameter:  
     
   CREATE STAGE azstage  
   URL = azure://<storageaccount>.blob.core.windows.net/tpch1000  
   CREDENTIALS=(AZURE\_SAS\_TOKEN=…)
3. Use the COPY statement to load data files from the stage into a target table (ORDERS in this example) in Snowflake:  
     
   COPY INTO Orders   
   FROM @azstage/orders

You can find more information about how to configure your Azure Blob Storage account for data loading with Snowflake in the following documentation page: <https://docs.snowflake.net/manuals/user-guide/data-load-azure-config.html>.

Consider using Azure Data Factory (<https://azure.microsoft.com/en-us/services/data-factory/>) for richer orchestration scenarios. Azure Data Factory helps with extracting data from multiple Azure services and persist the data as load files in Blob Storage.

You can use these steps to load the files with the order processing data from Azure Blob Storage. For this example, we have been using TPCH, a common data set, and Figure 3 shows the blob storage account with the data directories. You can use several COPY statements like the one above to populate the order processing data in your Snowflake tables.

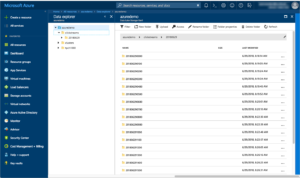


*Figure 3.*

**Azure Data Lake Store: Connecting Snowflake with Azure HDInsight, Spark and Azure Data Lake Store**

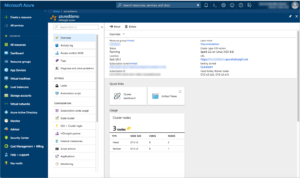
Many customers rely on Apache Spark as an integral part of their data analytics solutions. Snowflake natively integrates with Spark through its Spark connector. Below you can see how to use Spark in Azure HDInsight together with Snowflake. The use case implements a data pipeline originating from data stored in Azure Data Lake Store via HDInsight Spark into a Snowflake table.

Our running example will use this approach to make the clickstream data available in Snowflake next to the order processing data. The following screenshot (Figure 4) shows the folders with load files for the clickstream data in Azure Data Lake Store.



*Figure 4.*

The HDInsight cluster shown in Figure 5 below runs Spark 2.2. The cluster has been configured with a service principal so that it can access the clickstream files in the data lake account.



*Figure 5.*

Using the built-in Jupyter notebook capability of HDInsight, you can now run the following simple PySpark program to populate a dataframe in Spark with the clickstream data from data lake store and then persist the dataframe into a Snowflake table:

%configure  
{ “conf”: {“spark.jars.packages”: “net.snowflake:spark-snowflake\_2.11:2.4.0-spark\_2.2” }}

df = spark.read.csv(“adl://azuredemo.azuredatalakestore.net/clickstreams/20180629/\*”, header=“true”)

sfOptions = {  
 “sfURL”: “<sfaccount>.east-us-2.azure.snowflakecomputing.com”,  
 “sfAccount”: “<sfaccount>”,  
 “sfUser”: “<user>”,  
“sfPassword”: “<password>”,  
“sfDatabase”: “<dbname>”,  
“sfSchema”: “<schemaname>”,  
“sfWarehouse”: “<whname>”  
}

SNOWFLAKE\_SOURCE\_NAME = “net.snowflake.spark.snowflake”

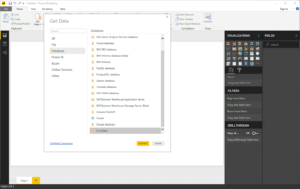
df.write.format(SNOWFLAKE\_SOURCE\_NAME).options(\*\*sfOptions).option(“dbtable”, “CLICKSTREAM”).mode(“append”).save()

Note the use of the %configure magic in the first lines of the code snippet. This loads the Snowflake Spark Connector that provides deep integration between Spark and Snowflake for highly optimized performance. After loading the clickstream data into the dataframe *df*, you can perform further transformations in Spark before writing the result into a Snowflake table. The last line of the code shows how to append data from the Spark dataframe to a table called CLICKSTREAM in Snowflake.

Now that we have loaded both the transactional data and the clickstream logs into Snowflake, we are ready to start analyzing the data sets. The next paragraph explains how to do that with Snowflake on Azure and Microsoft PowerBI.

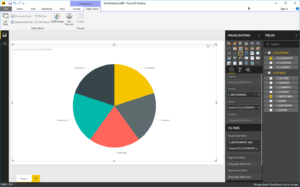
**PowerBI: Visualizing your Snowflake Analytics**

You can connect PowerBI Desktop to Snowflake by installing the Snowflake ODBC driver on your PowerBI Desktop machine. Once installed, you can use the built-in Snowflake data source for PowerBI. Click on the ‘Get Data’ button in PowerBI and then choose the ‘Database’ section, as shown in Figure 6. You can find Snowflake at the bottom of the list of supported data sources.



*Figure 6.*

After connecting to Snowflake using the hostname from your Snowflake account URL, you can explore the tables and develop powerful reports and charts. In Figure 7 below, you can see a chart created from both the transactional and the clickstream data sets. It shows the distribution of daily online customer activity from the clickstream logs over customer market segments from the order processing data.



*Figure 7.*

You can also publish your Snowflake reports to the PowerBI service. Note that you need the PowerBI gateway to connect PowerBI service to Snowflake.

This completes our running example for Snowflake on Azure. As part of this scenario, we have seen how to load data from both Azure Blob Storage and Azure Data Lake Store into Snowflake on Azure. We also demonstrated how to use Microsoft PowerBI to visualize your analytics in Snowflake.

**Resources**  
Webinar: [**Snowflake on Azure: Modern Data Analytics, August 23, 11am PT**](https://www.snowflake.com/webinar/snowflake-on-azure-modern-data-analytics/?utm_source=blog&utm_medium=referral&utm_campaign=azure-webinar-torsten-blog)  
Blog: [**Providing Customer Choice: Snowflake on Azure**](https://www.snowflake.com/blog/providing-customer-choice-snowflake-on-azure/)  
Partner Solutions: [**Snowflake on Microsoft Azure**](http://snowflake.net/technology-partners/microsoft/)