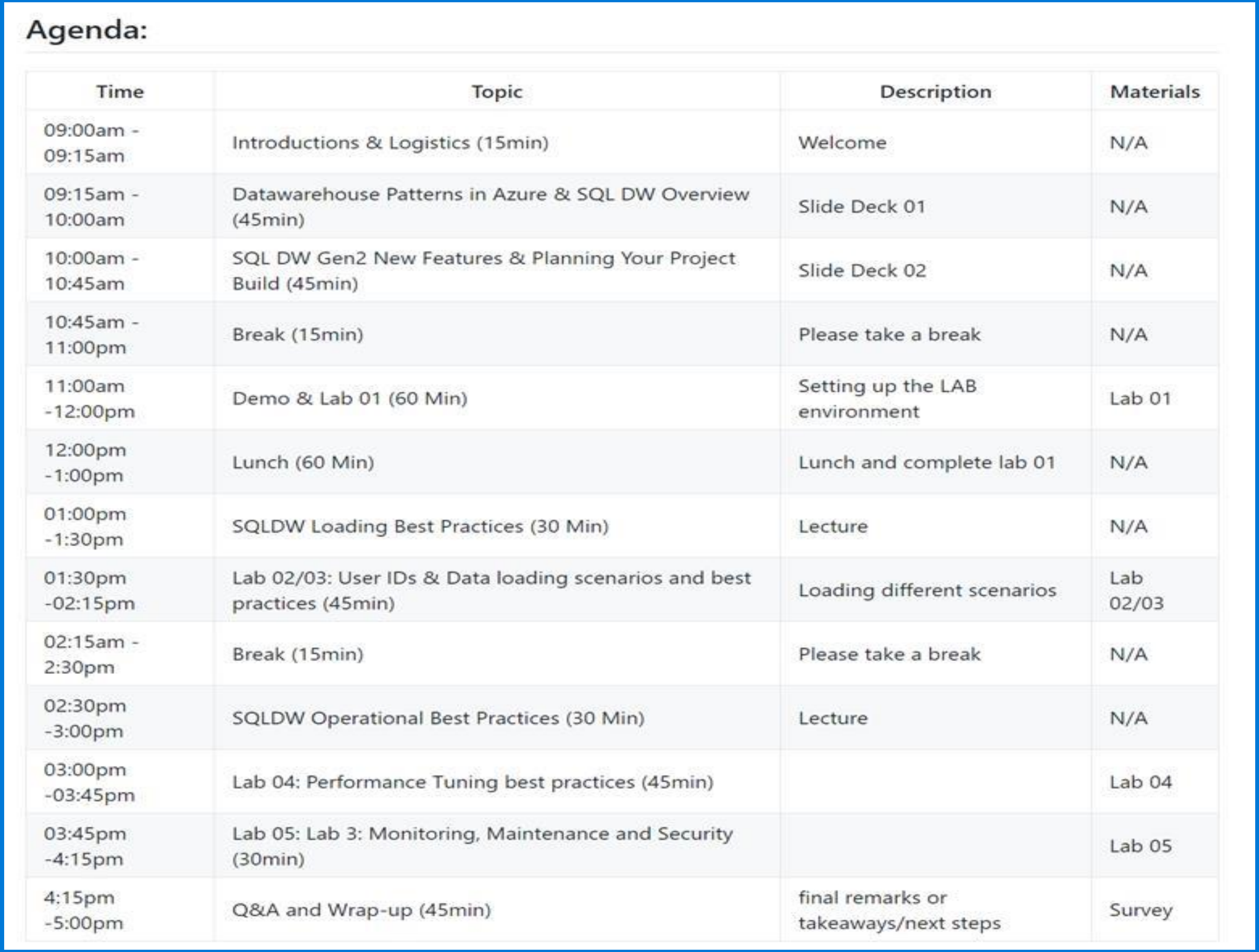
03.SQL DW Data Loading Best Practices

Microsoft

Azure Data Warehouse In-A-Day

Steve Young - Data & AI CSA
Rebecca Young - Data & AI CSA

Agenda



Agenda:

Time	Topic	Description	Materials
09:00am - 09:15am	Introductions & Logistics (15min)	Welcome	N/A
09:15am - 10:00am	Datawarehouse Patterns in Azure & SQL DW Overview (45min)	Slide Deck 01	N/A
10:00am - 10:45am	SQL DW Gen2 New Features & Planning Your Project Build (45min)	Slide Deck 02	N/A
10:45am - 11:00pm	Break (15min)	Please take a break	N/A
11:00am -12:00pm	Demo & Lab 01 (60 Min)	Setting up the LAB environment	Lab 01
12:00pm -1:00pm	Lunch (60 Min)	Lunch and complete lab 01	N/A
01:00pm -1:30pm	SQLDW Loading Best Practices (30 Min)	Lecture	N/A
01:30pm -02:15pm	Lab 02/03: User IDs & Data loading scenarios and best practices (45min)	Loading different scenarios	Lab 02/03
02:15am - 2:30pm	Break (15min)	Please take a break	N/A
02:30pm -3:00pm	SQLDW Operational Best Practices (30 Min)	Lecture	N/A
03:00pm -03:45pm	Lab 04: Performance Tuning best practices (45min)		Lab 04
03:45pm -4:15pm	Lab 05: Lab 3: Monitoring, Maintenance and Security (30min)		Lab 05
4:15pm -5:00pm	Q&A and Wrap-up (45min)	final remarks or takeaways/next steps	Survey

Microsoft Azure

Productive + Hybrid + Open + Trusted

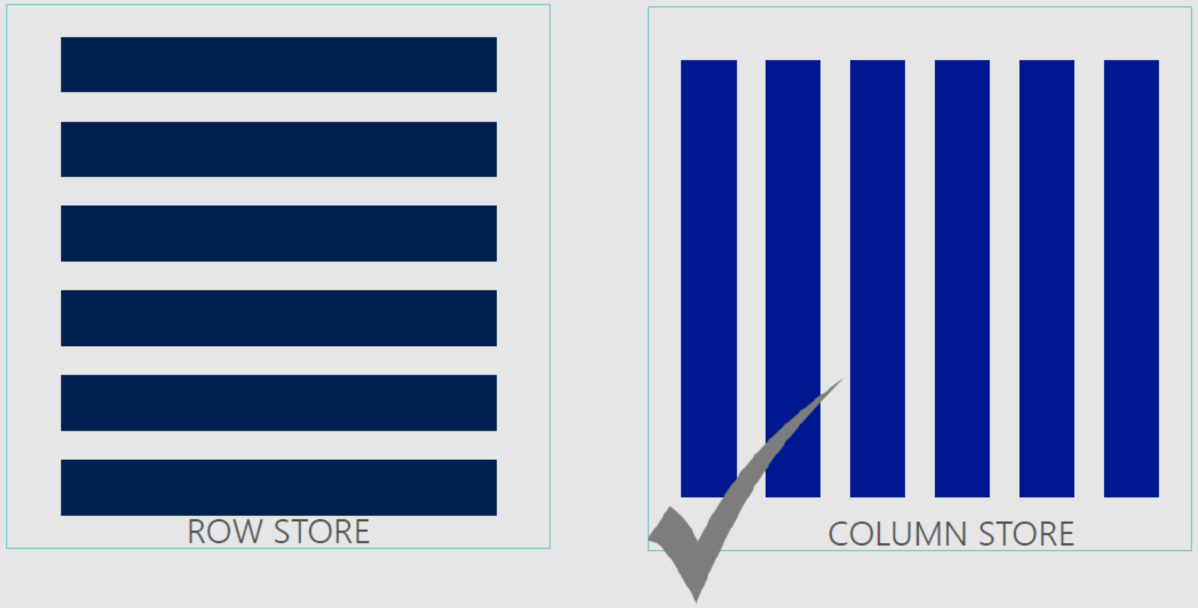
SQL DW Data Loading Best Practices

Topics

- Table Structure and impacts to loading
- Loading patterns
- Loading tools
- Loading best practices

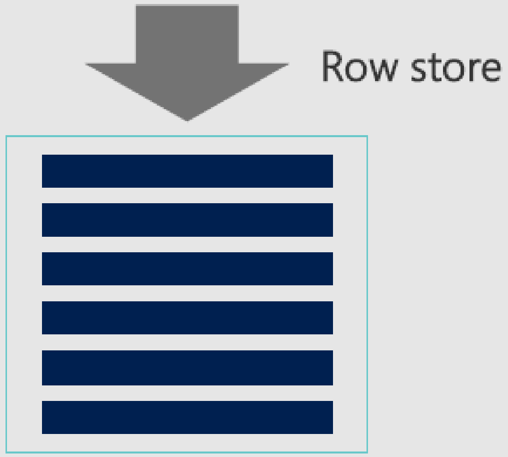
Physical Structures

Row store & Column Store



Row store or Column Store

- Small Data Set (< 60 million rows)
- Frequent updates
- Small Dimension tables



- Large Data Set (> 60 million rows)
- Mostly append only data
- Fact tables or large dimension tables

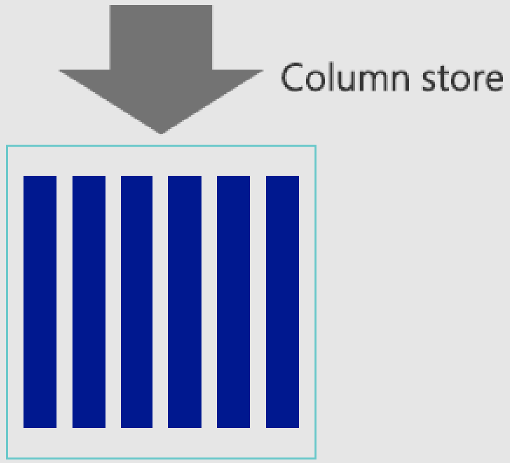


Table and Index Terminology

Primary Data Stores

Heap = Base Row Store

Clustered Index (CI) = Base Row Store maintained as a B-Tree

Clustered Columnstore Index (CCI) = Base Column Store

Why Partition

Benefit to Loads

Data Lifecycle Management

- Drop partition avoids transaction logging
- Insert to empty table/partition avoids transaction logging
- ⇒ Partition Switching pattern

Targeted Index Builds

Benefit to Queries

Partition Elimination

Partitioning Guidance

Partition for data management

Lesser benefit had on partition elimination for faster performance

Don’t over partition!

Partitioning granularity likely to differ to SQL Server

- Data is already spread across 60 distributions

Columnstore index row groups give ideal performance with 1 million rows each

Need at least 60 million per partition!

Loading Patterns

Loading Patterns

Pattern 1: Batch loads (PolyBase)

Load large volumes of data in parallel

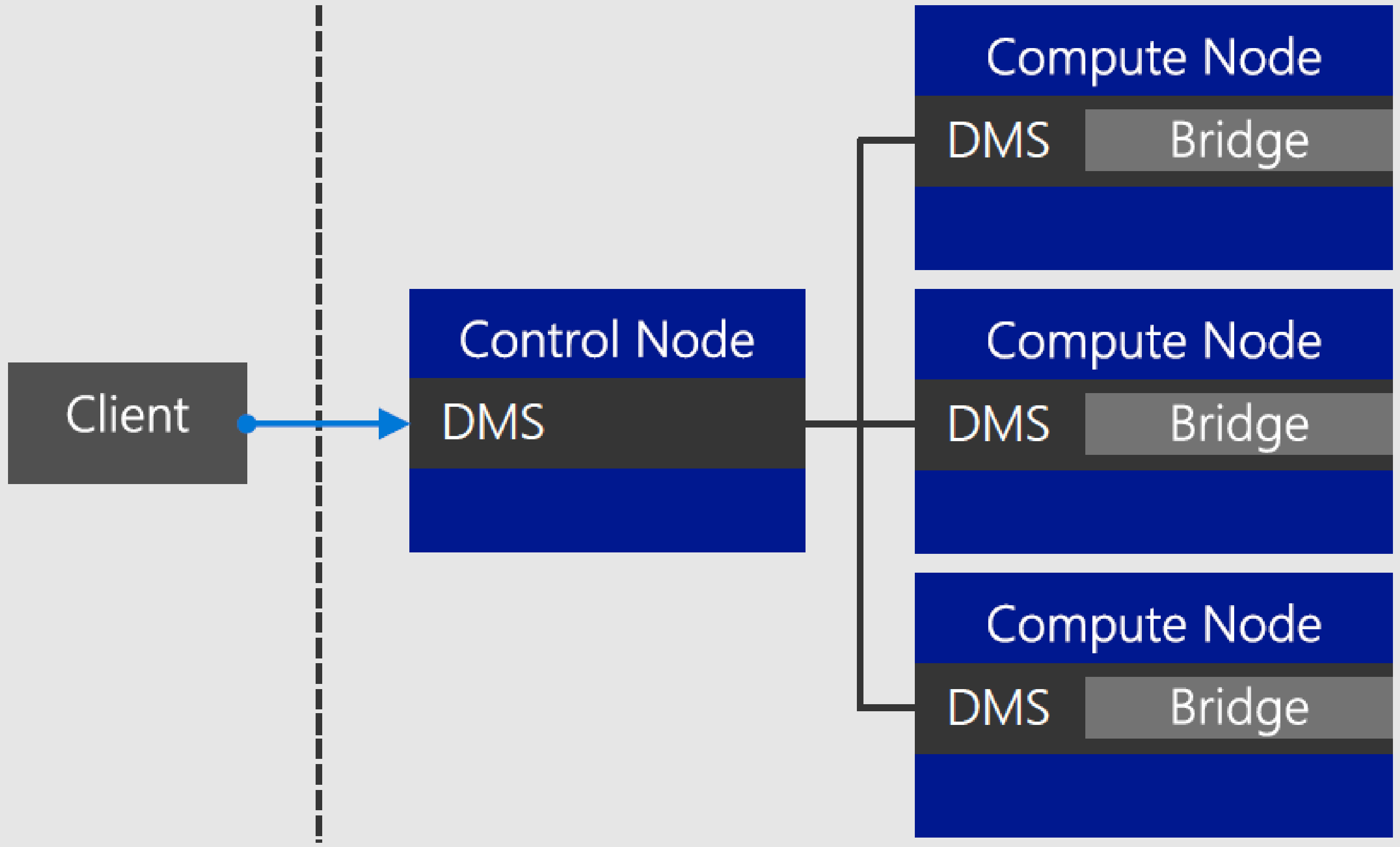
Move data from Azure storage to SQL DW in parallel to Compute nodes.

This includes Databricks integrations!

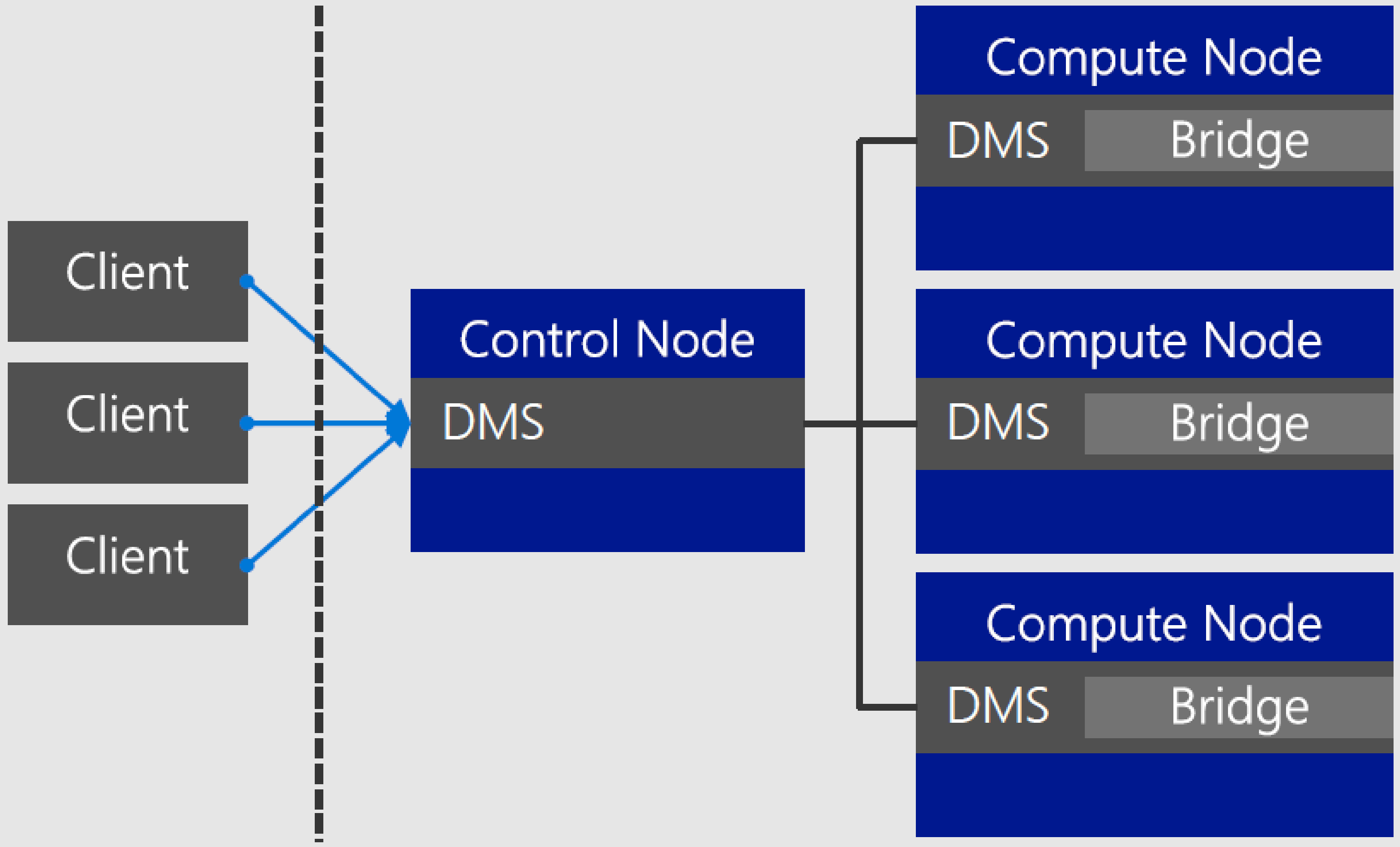
Pattern 2: Streaming loads (BCP)

- Single record or small batches in each load.
- Data moves from source to the Control node and then to the Compute nodes.

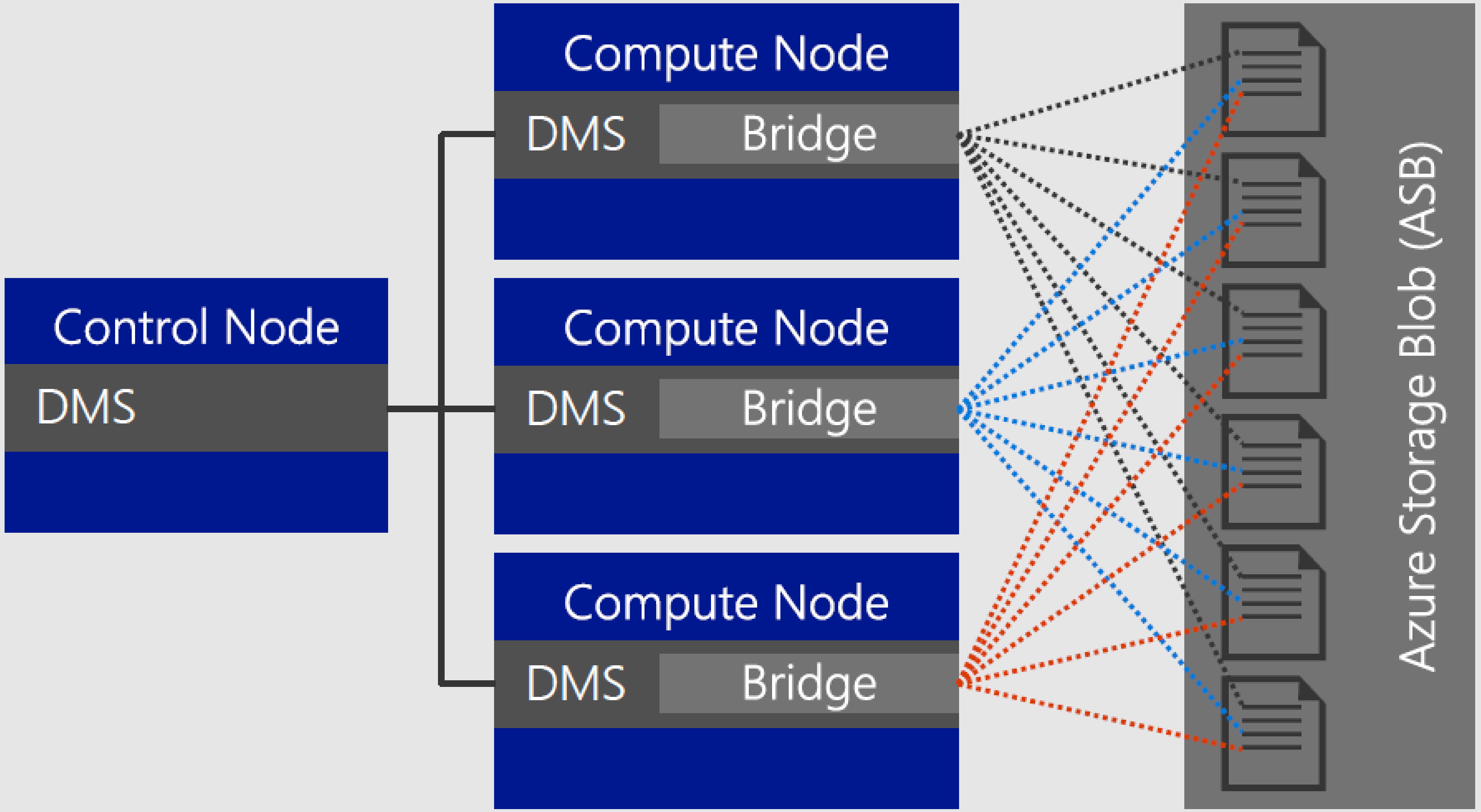
Single Gated Client



Single Gated Client Parallelized



Polybase parallel load to Azure Storage Blob



Loading Tools

Create External Tables

```
CREATE EXTERNAL DATA SOURCE WASBStor
WITH (TYPE = Hadoop,
      LOCATION = ‘wasbs://<container>@<account_name>.blob.core.windows.net',
          Credential = <Database scoped credential>);


CREATE EXTERNAL FILE FORMAT TextFile
WITH ( FORMAT_TYPE = DELIMITEDTEXT,
          DATA_COMPRESSION = 'org.apache.hadoop.io.compress.GzipCodec',
          FORMAT_OPTIONS (FIELD_TERMINATOR ='|', USE_TYPE_DEFAULT = TRUE));


CREATE EXTERNAL TABLE [dbo].[Customer_import] (
    [SensorKey] int NOT NULL,
    [CustomerKey] int NOT NULL,
    [Speed] float NOT NULL
)
WITH (LOCATION=‘/Dimensions/customer',
      DATA_SOURCE = WASBStor,
      FILE_FORMAT = TextFile
)
```

Once per WASB container

Once per file format

File path

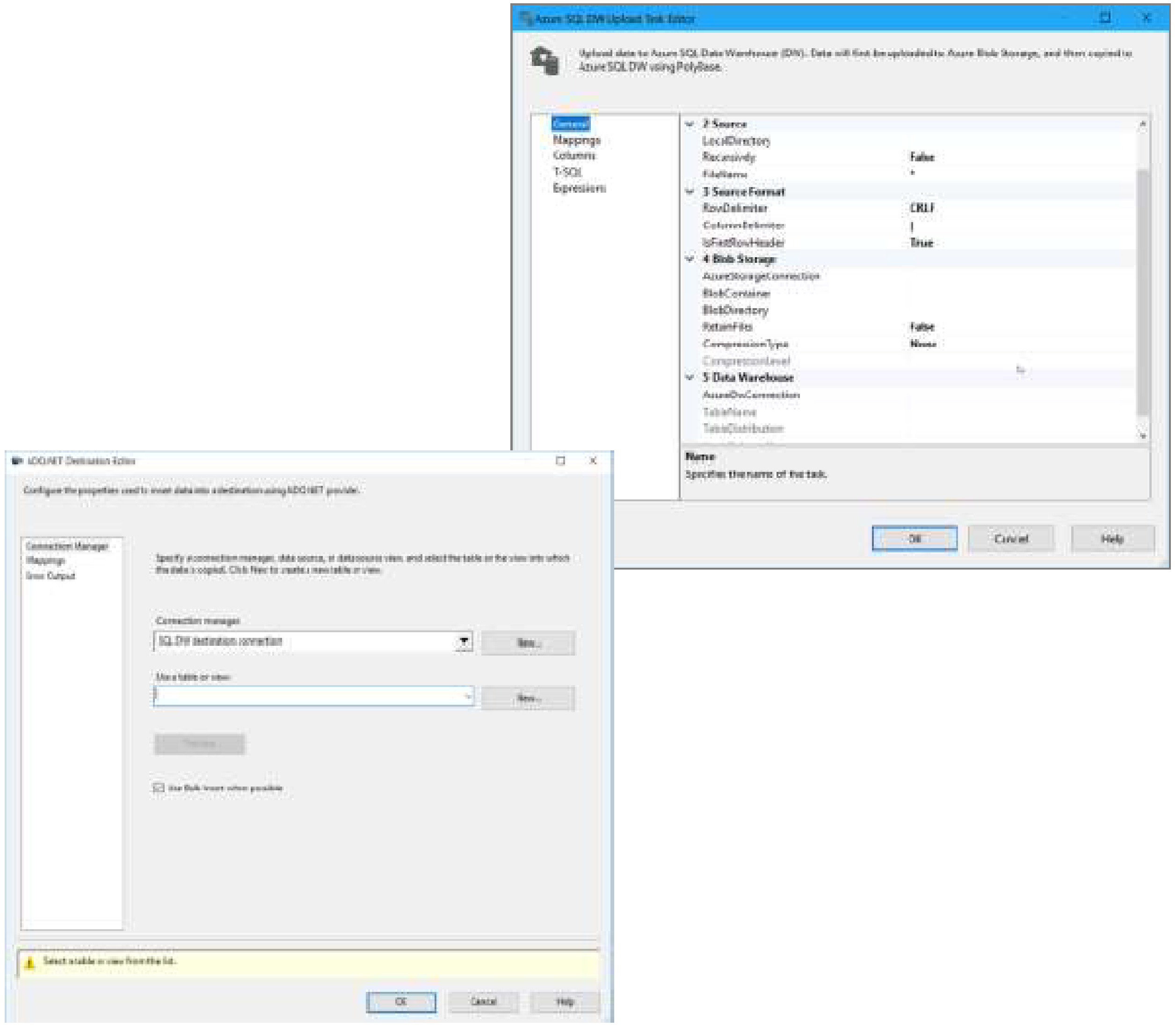
SQL Server Integration Services (SSIS)

Overview

SQL Server Integration Services is used to extract, transform data and load data from a variety of sources into Azure SQL Data Warehouse.

There are two options for loading data into SQL Data Warehouse with SSIS:

- **Azure SQL Data Warehouse Upload Task**: provides best performance but assumes source data is in delimited text file format.
- **Data Flow Task**: slower than SQL Data Warehouse Upload Task but supports a wider range of data sources.



Azure SQL DW Upload Task Editor

Upload data to Azure SQL Data Warehouse (DW). Data will first be uploaded to Azure Blob Storage, and then copied to Azure SQL DW using PolyBase.

General
Mappings
Columns
T-SQL
Expressions

2 Source	
LocalDirectory	
Recursively	False
FileName	*
3 Source Format	
RowDelimiter	CRLF
ColumnDelimiter	|
IsFirstRowHeader	True
4 Blob Storage	
AzureStorageConnection	
BlobContainer	
BlobDirectory	
RetainFiles	False
CompressionType	None
CompressionLevel	
5 Data Warehouse	
AzureDwConnection	
TableName	
TableDistribution	

Name
Specifies the name of the task.

ADO.NET Destination Editor

Configure the properties used to insert data into a destination using ADO.NET provider.

Connection Manager
Mappings
Error Output

Specify a connection manager, data source, or data source view, and select the table or the view into which the data is copied. Click New to create a new table or view.

Connection manager: SQL DW destination connection

Use a table or view:

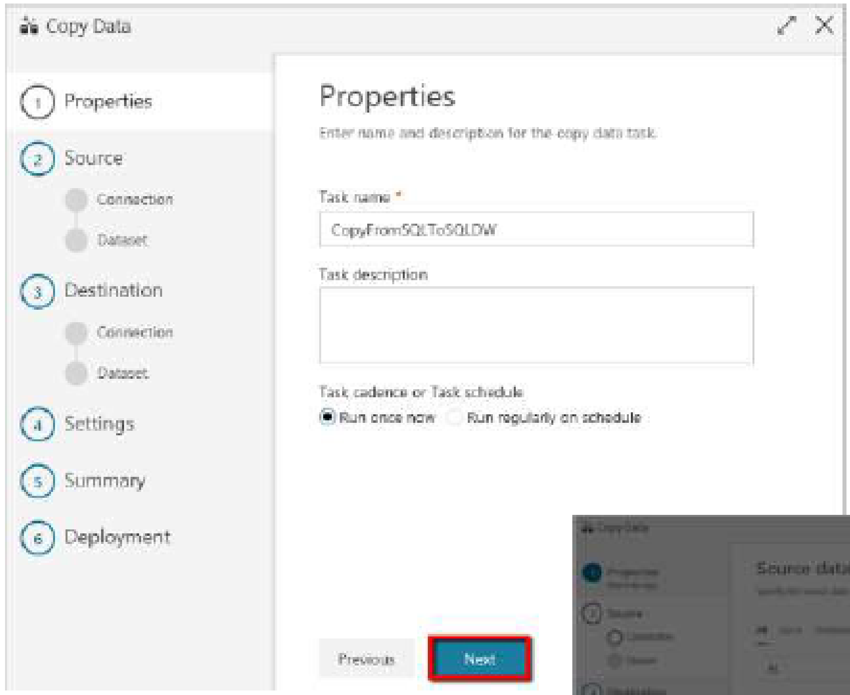
Use Bulk Insert when possible

Select a table or view from the list.

Azure Data Factory Copy Data Tool

Overview

The Azure Data Factory Copy Data tool provides an intuitive wizard that allows you to copy data from a variety of data sources into Azure SQL Data Warehouse.


Copy Data

1 Properties
2 Source
Connection
Dataset
3 Destination
Connection
Dataset
4 Settings
5 Summary
6 Deployment

Properties

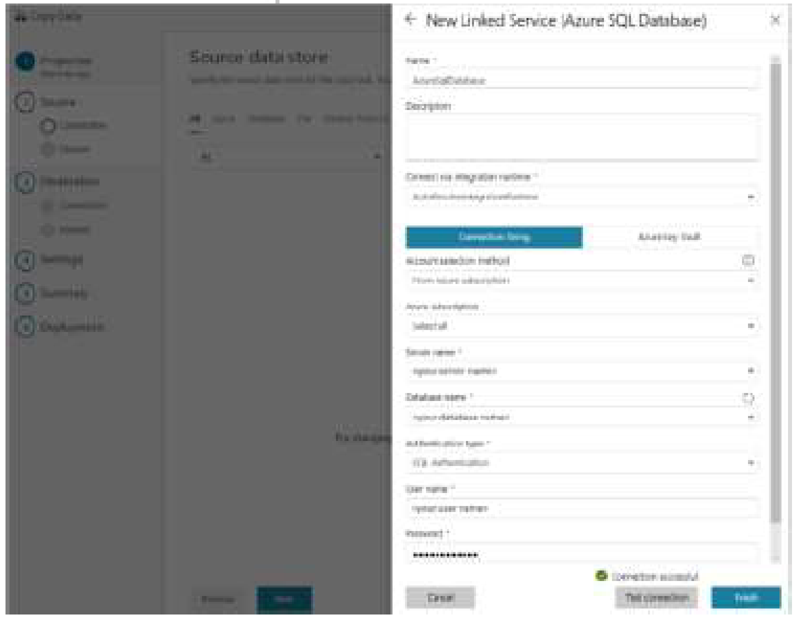
Enter name and description for the copy data task.

Task name *
CopyFromSQLToSQLDW

Task description

Task cadence or Task schedule
[✓ Run once now] Run regularly on schedule

Previous Next



New Linked Service (Azure SQL Database)

Name *
AzureSqlDatabase

Description

Connect via integration runtime *
[illegible]

[✓ Connection String] Azure Key Vault

Account selection method
From Azure subscription

Azure subscription
Select all

Server name *
<your server name>

Database name *
<your database name>

Authentication type *
SQL Authentication

User name *
<your user name>

Password *
•••••••••••

Connection successful

Cancel Test connection Finish

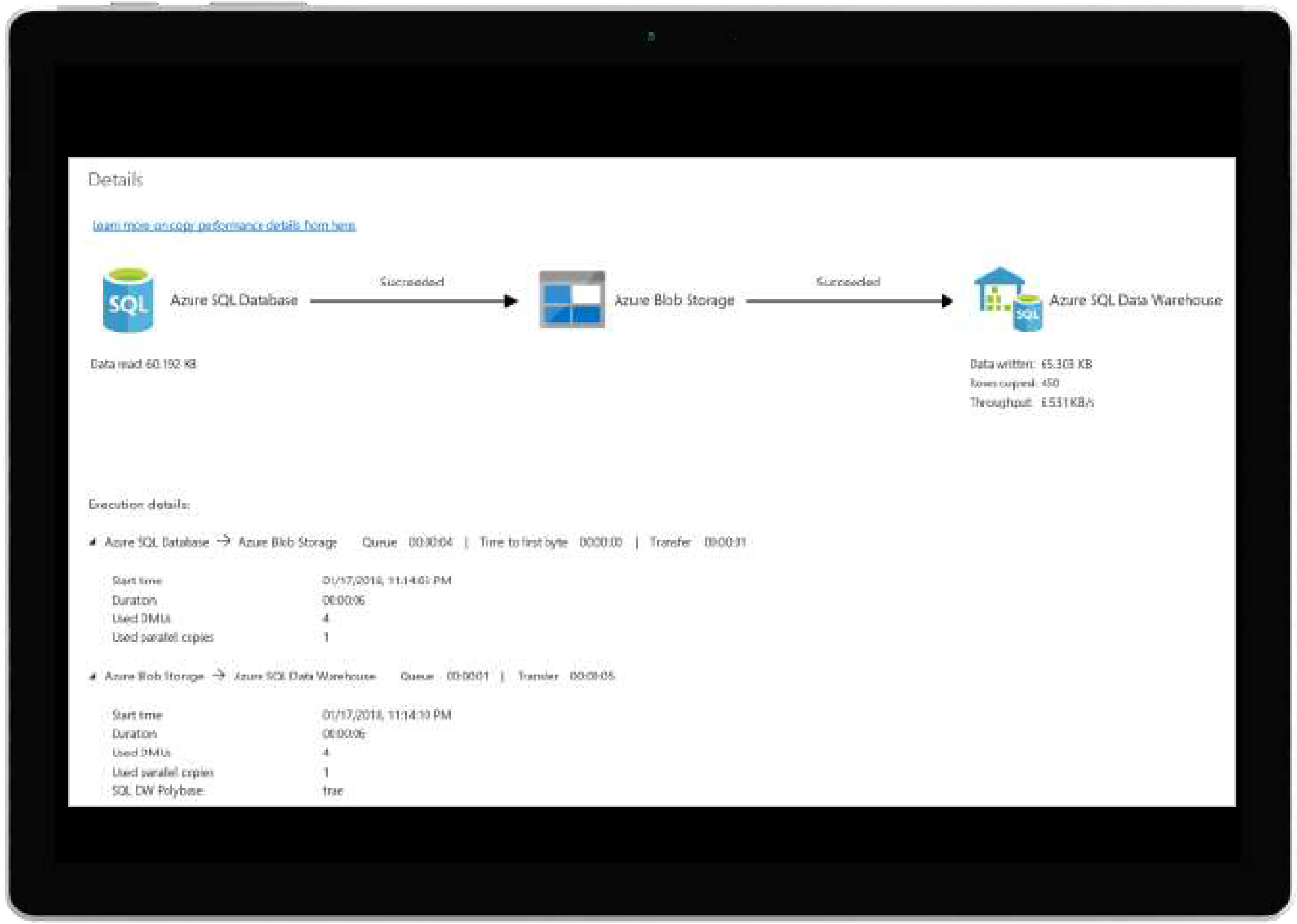
Azure Data Factory Copy Activity

Overview

The Azure Data Factory Copy activity allows copying to and from Azure SQL Data Warehouse from any supported data store.

The Copy activity also supports retrieving data from a SQL source by using a SQL query or stored procedure. Authentication can be via:

- SQL Authentication
- Service principal token authentication
- Managed identity token authentication



Details

learn more on copy performance details from here

Data read: 60.192 KB

Data written: 65.303 KB
Rows copied: 450
Throughput: 6.531 KB/s

Execution details:

```
◢ Azure SQL Database → Azure Blob Storage    Queue  00:00:04  |  Time to first byte  00:00:00  |  Transfer  00:00:11

    Start time               01/17/2018, 11:14:03 PM
    Duration                 00:00:06
    Used DMUs                4
    Used parallel copies     1

◢ Azure Blob Storage → Azure SQL Data Warehouse    Queue  00:00:01  |  Transfer  00:00:05

    Start time               01/17/2018, 11:14:10 PM
    Duration                 00:00:06
    Used DMUs                4
    Used parallel copies     1
    SQL DW Polybase          true
```

Data Bricks Streaming

Overview

The Databricks SQL DW connector supports batch and structured streaming support for writing real-time data into Azure SQL Data Warehouse.

It uses Polybase and the Databricks structured streaming API to stream data from Kafka or Kinesis sources directly into SQL Data Warehouse at a user-configurable rate.

Source: https://docs.azuredatabricks.net/spark/latest/data-sources/azure/sql-data-warehouse.html#streaming-support

```
# Prepare streaming source; this could be Kafka,
Kinesis, or a simple rate stream.
df = spark.readStream \
  .format("rate") \
  .option("rowsPerSecond", "100000") \
  .option("numPartitions", "16") \
  .load()

# Apply some transformations to the data then use
# Structured Streaming API to continuously write the
data to a table in SQL DW.
df.writeStream \
  .format("com.databricks.spark.sqldw") \
  .option("url", <azure-sqldw-jdbc-url>) \
  .option("tempDir",
"wasbs://<containername>@<storageaccount>.blob.core.
windows.net/<directory>") \
  .option("forwardSparkAzureStorageCredentials",
"true") \
  .option("dbTable", <table-name>) \
  .option("checkpointLocation", "/tmp_location") \
  .start()
```

Mechanism for Loading

1. PolyBase
2. SSIS*
3. ADF
4. BCP
5. SQLBulkCopy API
6. Attunity Cloudbeam
7. ASA/Storm**

	PolyBase	SSIS	ADF	BCP	SqlBulkCopy
Rate	Fastest ↔ Slowest				
Rate increase as DWU increases	Yes	Yes	Yes	No	No
Rate increases as you add concurrent load	No	No	No	Yes	Yes

* With SSIS Azure Feature Pack June 2017 or newer

** Not a good idea

Loading Method Considerations

Loading Type	Source	Concerns	Advice
Batch loading	WASB/ADLS	Latency of data	Do it!
Micro Batch loading	WASB/ADLS	Potential impact to index health. Impact on machine resources	Do it with caution… Make sure that loads are big enough
Streaming Load	Azure Stream Analytics, BCP	Column store Index health. Load Performance	Do it with higher caution… Understand higher latency and impact on segment quality.

Best Practices

Dimension Tables – Best Practices

Use **round robin** or **replicated** for small tables

Use clustered index, not clustered columnstore index

Can load directly to production because of small size

Use metadata rename to reload data

Fact Tables Best Practices

Use partitions to reduce the loading impact on the production table

Consider landing data from ADL in a staging table

Take advantage of directory structure to limit loading scope

Additional Resources

- Guidance for designing distributed tables
- https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-tables-distribute

- Columnstore indexes
- https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-overview?view=sql-server-2017

- Analyze your workload in Azure SQL Data Warehouse
- https://docs.microsoft.com/en-us/azure/sql-data-warehouse/analyze-your-workload
- Adaptive caching powers Azure SQL Data Warehouse performance gains
- https://azure.microsoft.com/en-us/blog/adaptive-caching-powers-azure-sql-data-warehouse-performance-gains/

- Cheat sheet
- https://docs.microsoft.com/en-us/azure/sql-data-warehouse/cheat-sheet

TBD – Replace with links

Microsoft

Q&A