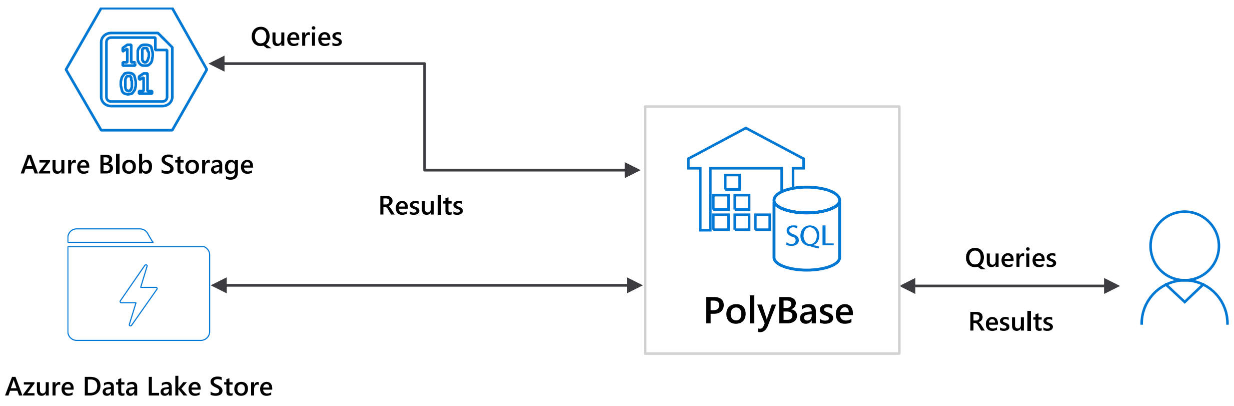
Extensive data ingestion capabilities

Polybase

Overview

Polybase enables users to run queries and import/export data residing outside of the SQL data warehouse.

Users write T-SQL to read and access data that is stored externally to the data warehouse.



Polybase – authentication

Overview

The credentials used for Polybase queries are stored in the data warehouse

Credentials encrypted with a symmetric key entered by the user during setup.

```
-- Create master key to encrypt credential secret

CREATE MASTER KEY ENCRYPTION BY PASSWORD =
'S0me!nfo';


-- Create database scoped credential for Azure Data
Lake Store Gen2 (required if not using managed
service identity)

CREATE DATABASE SCOPED CREDENTIAL
AzureStorageCredential

WITH IDENTITY = 'user', Secret =
'<storage_account_key>’;
```

SQL Data Warehouse managed identities

Overview

Azure SQL Data Warehouse supports managed service identity authentication to Azure Data Lake. This removes the need for storing access credentials in code or in Azure Key vault.

Managed identities can currently only be created via Azure PowerShell or Azure CLI.

```
 # Generate and assign an Azure AD Identity for DW
Set-AzureRmSqlServer -ResourceGroupName
$resourceGroupName -ServerName $serverName -
AssignIdentity


# Get ServicePrincipalId assigned to DW
$serverAzureAdIdentity =
(Get-AzureRmADServicePrincipal -SearchString
$serverName).Id
```

Polybase – external data sources

Overview

Polybase supports querying files stored in a Hadoop File System (HDFS), Azure Blob storage, or Azure Data Lake Store.

To query files, users create three objects: External data source, external file format, external table.

```
-- Create Azure DataLake Gen2 Storage reference
CREATE EXTERNAL DATA SOURCE AzureStorage with
(
TYPE = HADOOP,
LOCATION='abfss://<container>@<storageaccnt>.blob.core.windows.net',
CREDENTIAL = AzureStorageCredential -- not required if using
managed identity
);
-- Type of format in Hadoop (CSV, RCFILE , ORC, PARQUET).
CREATE EXTERNAL FILE FORMAT TextFileFormat WITH
(
FORMAT_TYPE = DELIMITEDTEXT,
FORMAT_OPTIONS (FIELD_TERMINATOR ='|', USE_TYPE_DEFAULT =
TRUE)
)
-- LOCATION: path to file or directory that contains data
CREATE EXTERNAL TABLE [dbo].[CarSensor_Data]
(
[SensorKey] int NOT NULL,
[Speed] float NOT NULL,
[YearMeasured] int NOT NULL
)
WITH (LOCATION='/Demo/’, DATA_SOURCE = AzureStorage,
FILE_FORMAT = TextFileFormat
);
```

Create table as select (CTAS)

Overview

The CTAS statement is a parallel operation that creates a new table based on the output of a SELECT statement.

Can also be used for column transformation

Ex: Change varchar size or transform datetime

Benefit

Simplest and fastest way to create a copy of a table.

Allows for ingesting data stored in an external source into managed data warehouse tables.

Redistribute tables for optimal performance (create hash-distributed or replicated tables)

```
-- Ingest external table data into data warehouse
CREATE TABLE [dbo].[FactInternetSales]
WITH
(
    DISTRIBUTION = ROUND_ROBIN
,   CLUSTERED COLUMNSTORE INDEX
)
AS
SELECT  *
FROM    [staging].[FactInternetSales]
;
```

Copy command

Overview

Simplifies loading data into data warehouse.

Accesses directly from external sources

- Data Lake Store Gen 2
- Blob Storage

Fully parallelized, scales with cluster compute.

No dependency on managed objects such as external tables.

```
--Create destination table in SQL DW
CREATE TABLE [dbo].[weatherTable]
(
  [ObservationTypeCode] [nvarchar](5) NOT NULL,
  [ObservationTypeName] [nvarchar](100) NOT NULL,
  [ObservationUnits] [nvarchar](5) NULL
)
WITH (
DISTRIBUTION = ROUND_ROBIN, HEAP);


--Copy files in parallel directly into datawarehouse table
COPY INTO [dbo].[weatherTable]
FROM 'abfss://<storageaccount>.blob.core.windows.net/<filepath>'
WITH (
FILE_FORMAT = 'DELIMITEDTEXT’,
SECRET = CredentialObject);
```

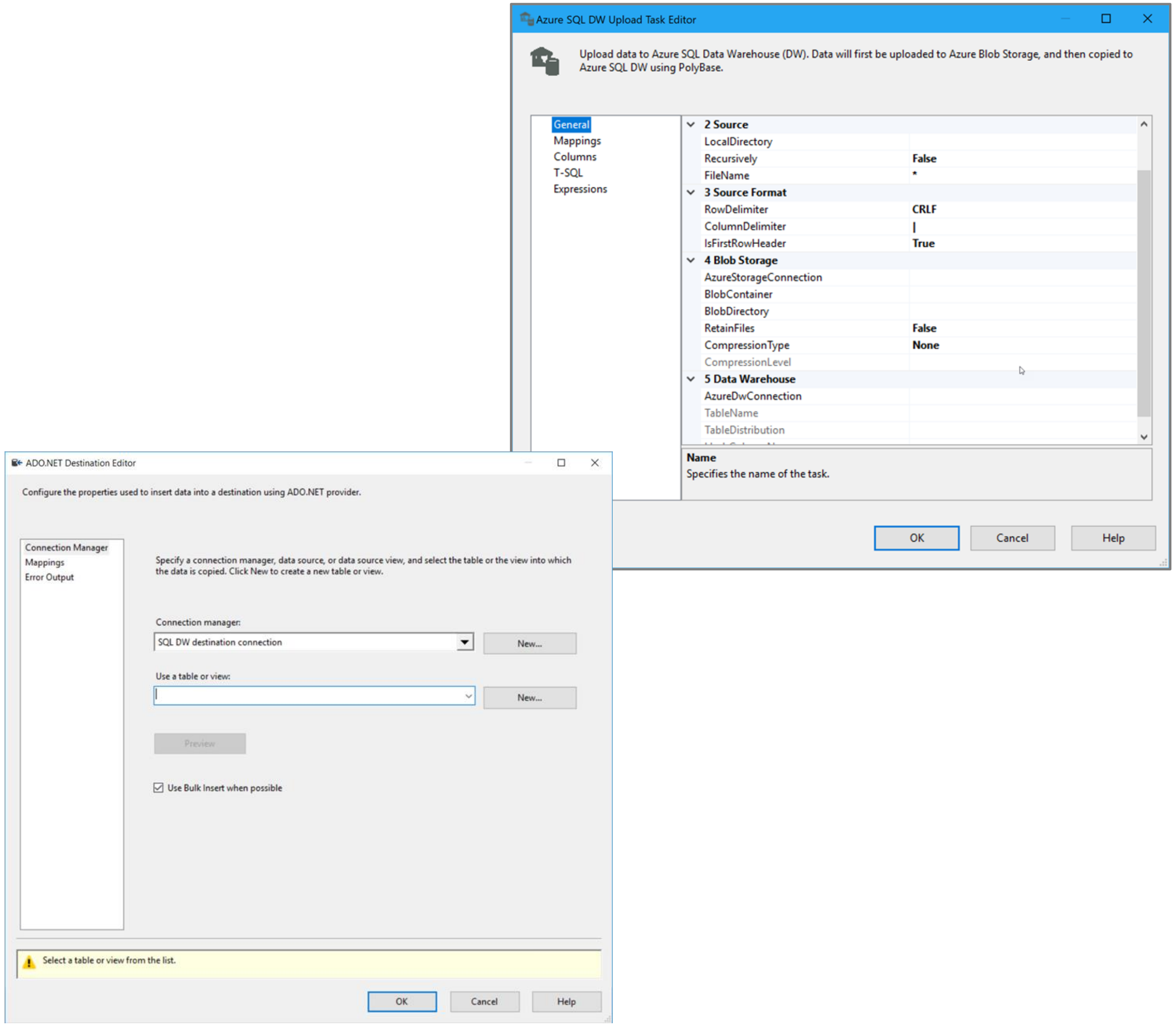
SQL Server Integration Services (SSIS)

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.

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.

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:

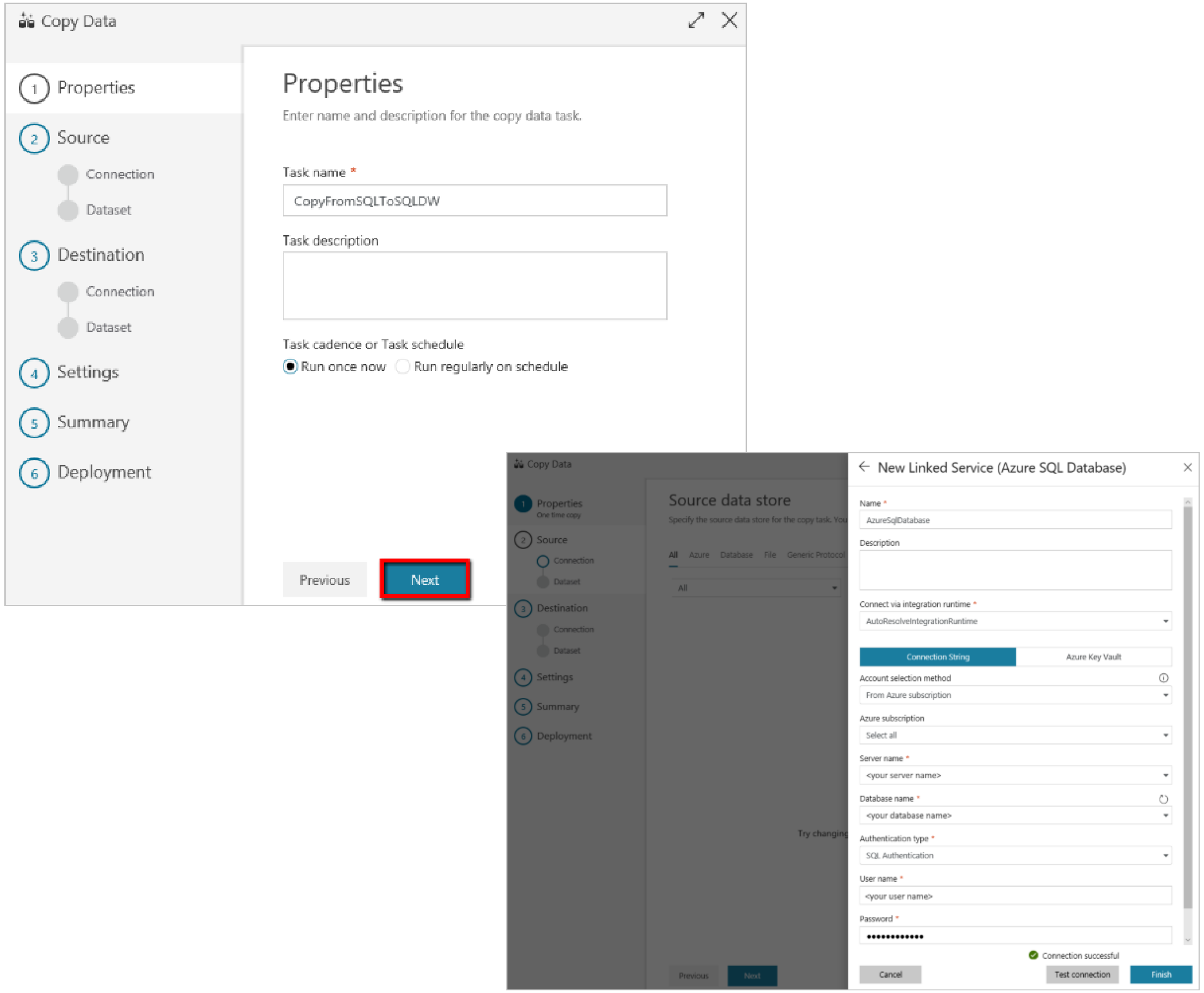
☑ Use Bulk Insert when possible

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.



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

Source data store

Specify the source data store for the copy task. You

All

Try changing

New Linked Service (Azure SQL Database)

Name * AzureSqlDatabase

Description

Connect via integration runtime * AutoResolveIntegrationRuntime

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

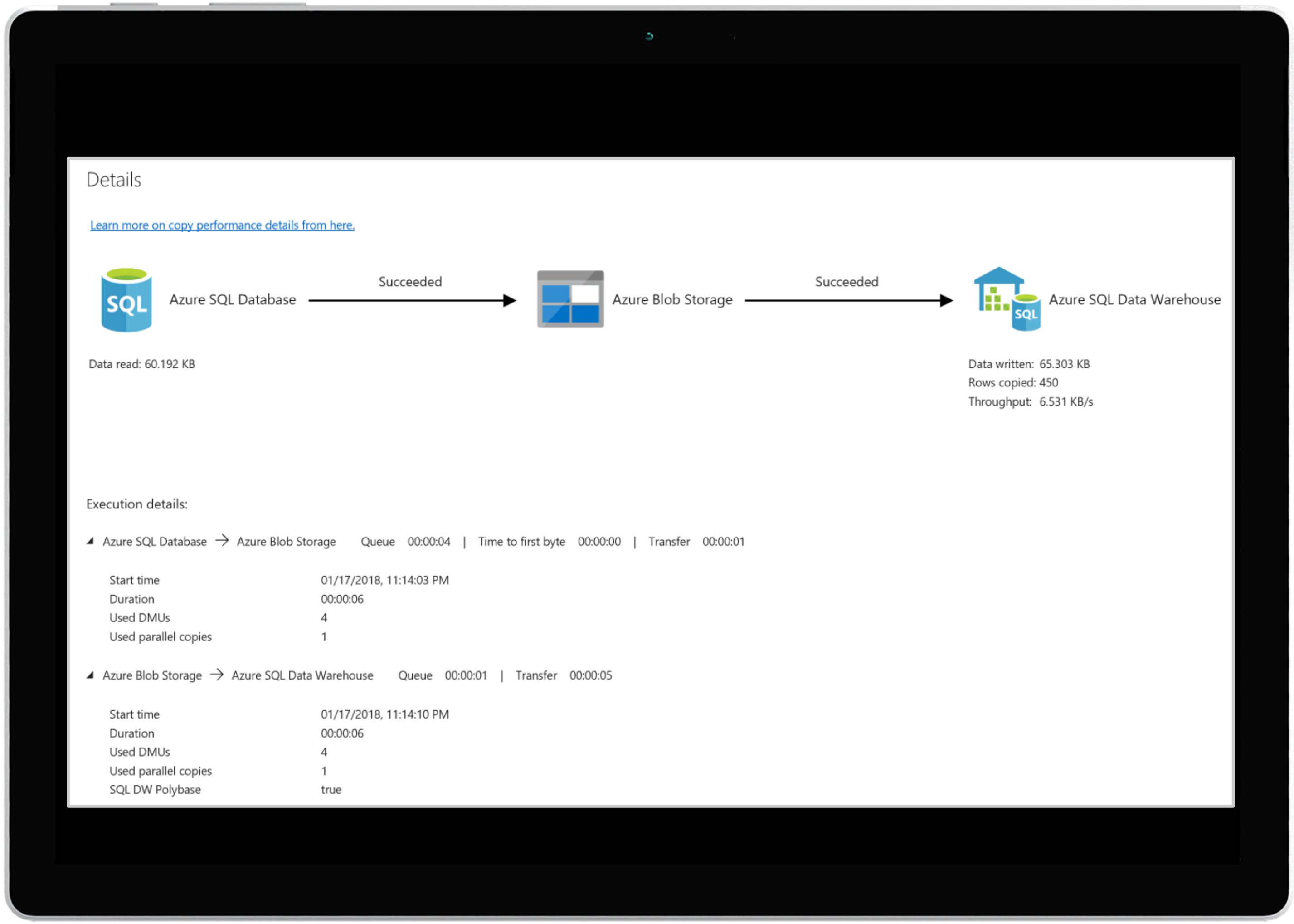
Azure Data Factory Copy activity

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   00:00:01

   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
```

Databricks – structured 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()
```

CETAS: Write and read to Azure Data Lake

Usage

Creates an external table and exports in parallel the results of a T-SQL query

Exports to Hadoop, Azure Blob Storage, or Azure Data Lake

After the CETAS statement finishes, you can run T-SQL queries on the external table

External table name and definition are stored as metadata.

```sql
-- Example is based on Adventure Works and writes
data to Azure Data Lake stores
CREATE EXTERNAL TABLE dbo.factInternetSalesExport
WITH
(
    LOCATION = '/files/Customer',
    DATA_SOURCE = customer_ds,
    FILE_FORMAT = customer_ff
)
AS SELECT T1.* FROM dbo.FactInternetSales T1 JOIN
dbo.DimCustomer T2
ON ( T1.CustomerKey = T2.CustomerKey )
OPTION ( HASH JOIN );
```

Microsoft

Databricks Demo