Azure SQL Data Warehouse

Data Loading and Export

Microsoft

Agenda

- Loading Options
- Loading Tools
- PolyBase
- DMVs
- Exporting to Azure Blob Storage

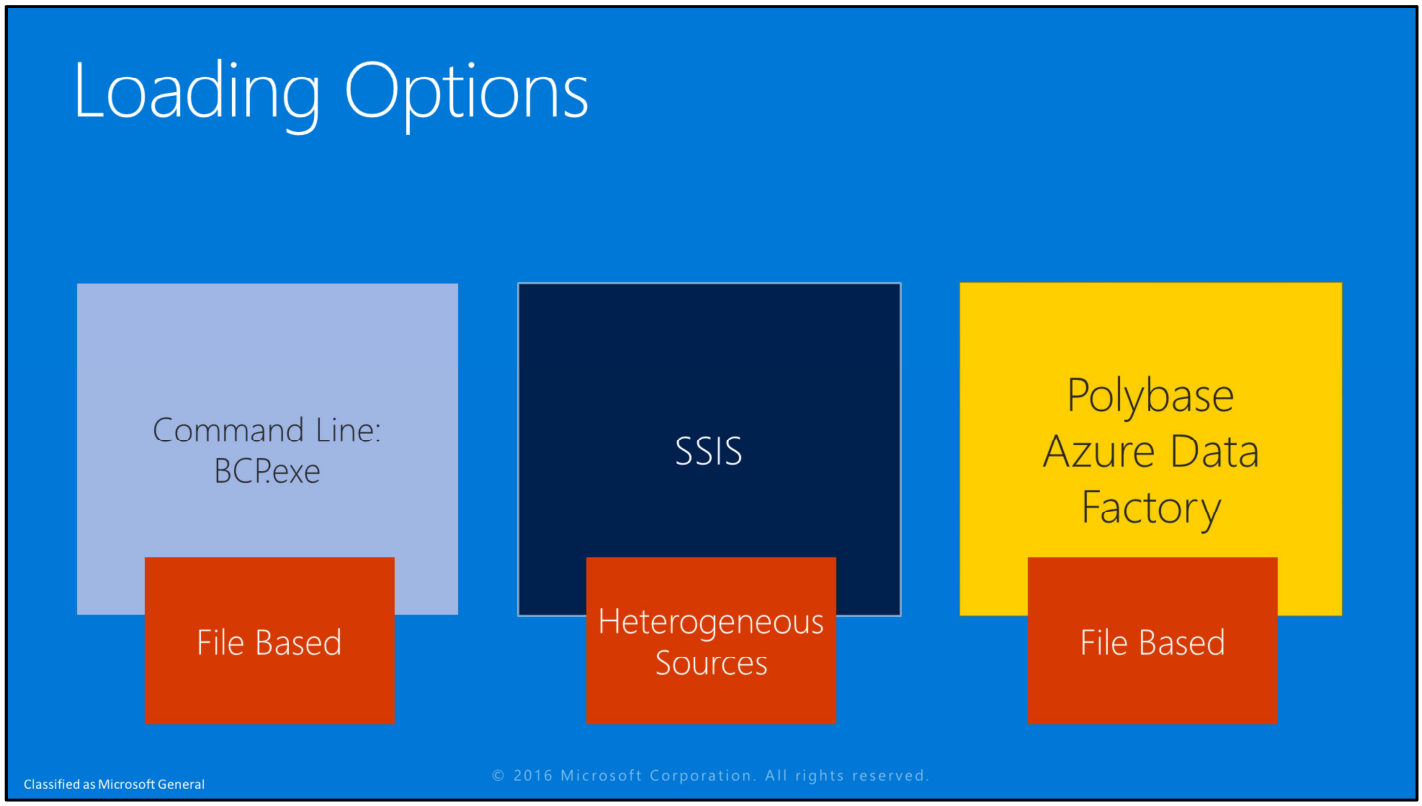
Agenda

- Loading Options
- Loading Tools
- PolyBase
- DMVs
- Exporting to Azure Blob Storage

Data Loading Design Goals

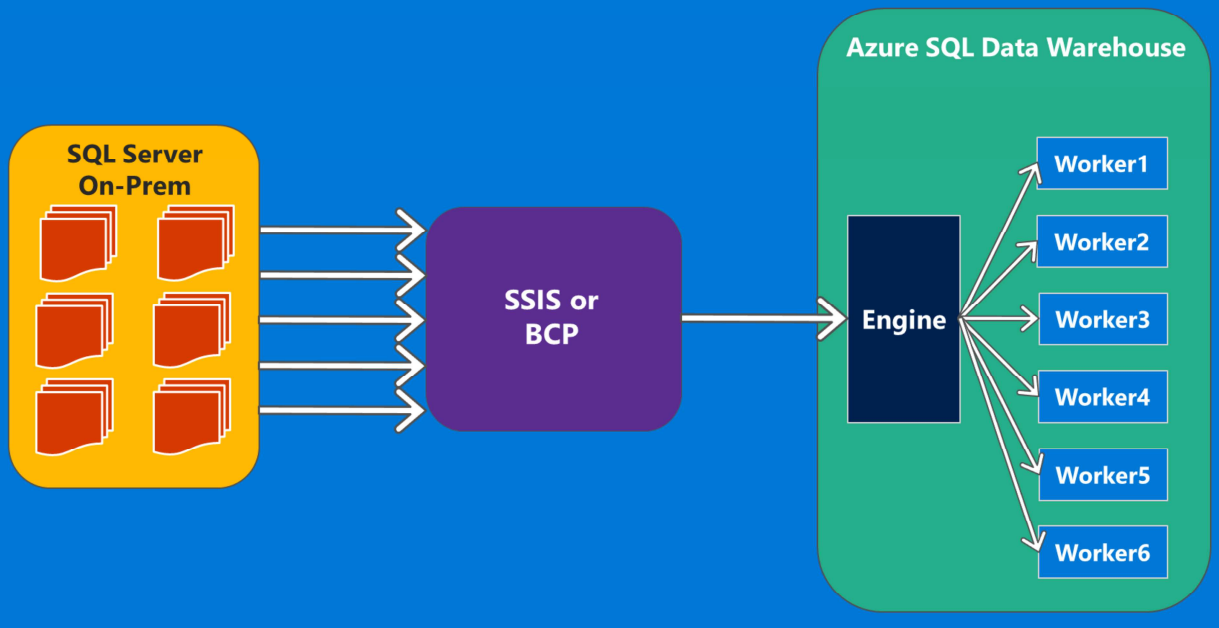
- Load data **efficiently**
- **Load Data non-obtrusively**, respecting concurrent queries and loads (not so much true for FASTAPPEND)
- **Reduce** table **fragmentation** as much as possible
- Provide system **recovery capabilities** in the event of data load failure with **minimal impact on concurrent queries**
- Provide **multiple load/ETL options** for customers

The SQL Server resource governor is used internally to do concurrency balancing between queries and data loads.

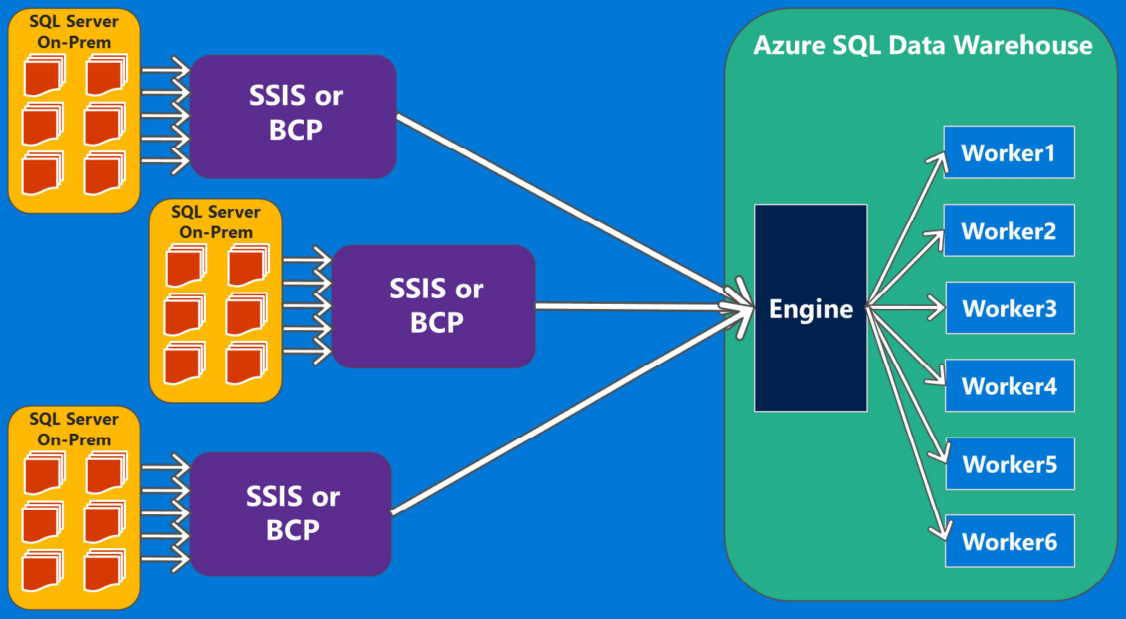


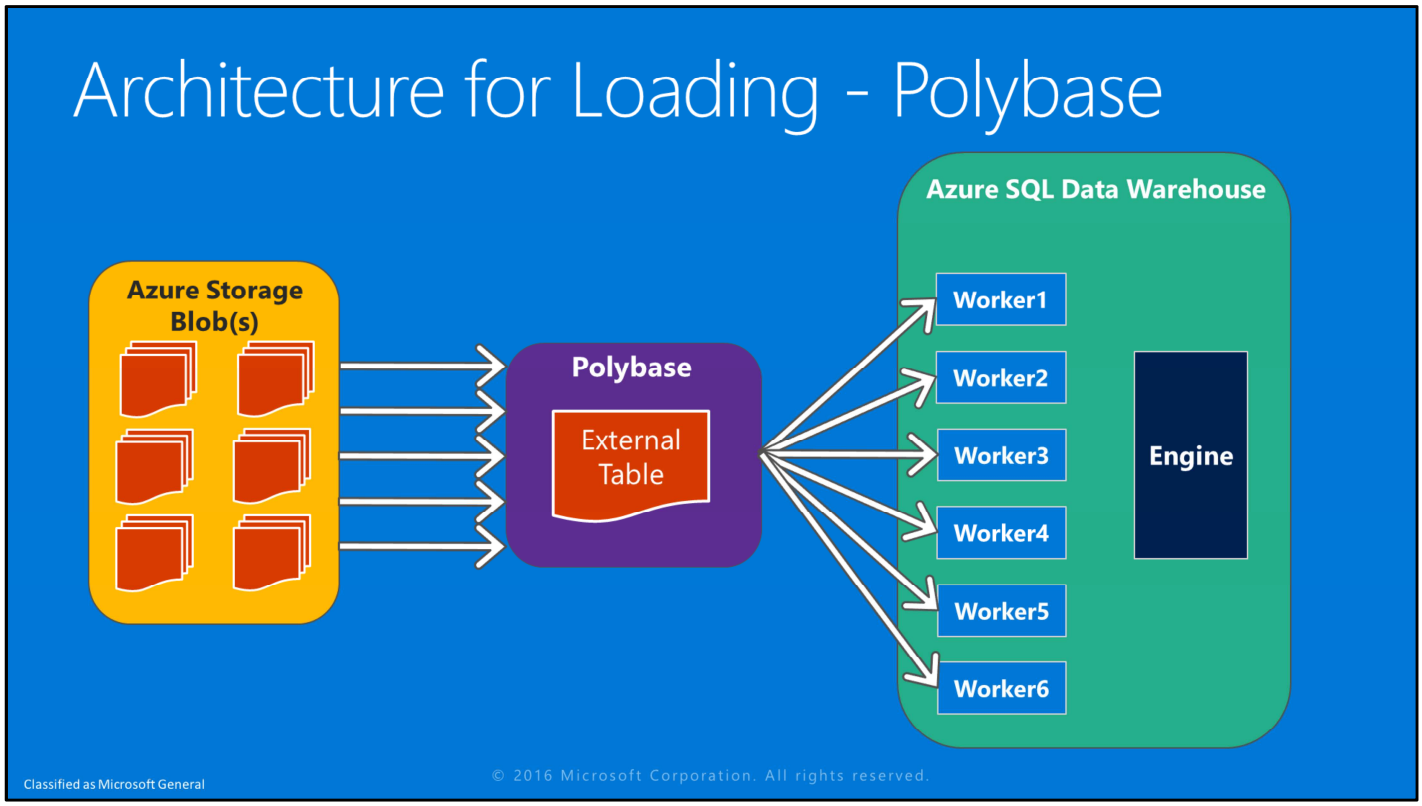
https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-overview-load/

Architecture for Loading – SSIS or BCP



Architecture for Loading – SSIS or BCP





https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-get-started-load-with-polybase/

Agenda

- Loading Options
- Loading Tools
- PolyBase
- DMVs
- Exporting to Azure Blob Storage

Loading ETL/ELT Options

BCP

SQL Server Integration Services (SSIS – ODBC Destination)

CREATE TABLE AS SELECT (CTAS)

Polybase

Third Party Tools

- For ODBC, use provider "SQL Server Native Client 11.0" or "ODBC Driver 11 for SQL Server"
- Alternatively you can also use the "OLEDB Destination" of provider "SQL Server Native Client 11.0"

bcp.exe – Parameters

BCP.exe { [database_name.][schema.] table_name | view_name | "query" }
{ in | out | queryout | format } file_name

Can Export Data from SQL DW

-a packet size
-b batch size
-c use character Type (don't prompt)
-C { ACP | OEM | RAW | code_page }
-d database name
-e error file
-E keep identity values in file
-F first_row
-h load "hints [,...n]"
-i input file name
-k keep null values
-K application intent
-L last_row
-l login time out

https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-load-with-bcp/

bcp.exe – Parameters

-m acceptable maximum errors
-n use native data types for import
-N keep non-text native
-o output file name
-P password
-q set quoted identifier on
-r row terminator
-R use regional format for data
-S [server_name[\instance_name]]
-f generates a text format file
-t field terminator
-T trusted connection
-U login name
-v version number of BCP.EXE
-V file format version
(80 | 90 | 100 | 110 | 120 | 130)
-w wide character type
-x generates a format file in XML
/? help

BCP data in

```
bcp {database}.dbo.lineitem_cci in
C:\Toolbox\dbgen\lineitem.tbl -c -U username
-P password -S xxxxxxxx.database.windows.net
-q -t"|" -r \n -w
```

SSIS and SQL DW Data Types

When using SSIS to load data from a data source to a SQL DW database:

- Data is first mapped from the source data to SSIS data types.
- This allows data from multiple data sources to map to a common set of data types.
- Then the data is mapped from SSIS to SQL DW data types.

https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-load-from-sql-server-with-integration-services/

SSIS and SQL DW Data Types Mapping

Examples of SSIS data types that can map to SQL DW data types:

SQL DW Data Type	SSIS Data Type(s) that can map to the SQL DW Data Type
INT	DT_I1, DT_I2, DT_I4, DT_UI1, DT_UI2
BIGINT	DT_I1, DT_I2, DT_I4, DT_I8, DT_UI1, DT_UI2, DT_UI4
CHAR	DT_STR
DATETIME	DT_DATE, DT_DBDATE, DT_DBTIMESTAMP, DT_DBTIMESTAMP2
DECIMAL	DT_DECIMAL, DT_I1, DT_I2, DT_I4, DT_I8, DT_NUMERIC, DT_UI1, DT_UI2, DT_UI4, DT_UI8

https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-tables-data-types/

SSIS

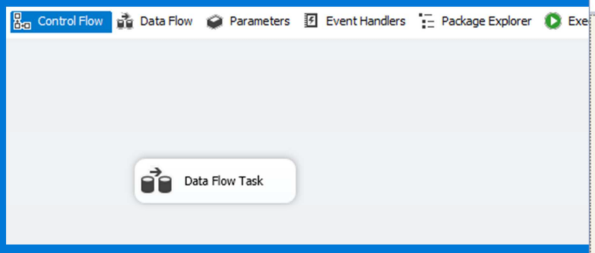
Destination Adaptors

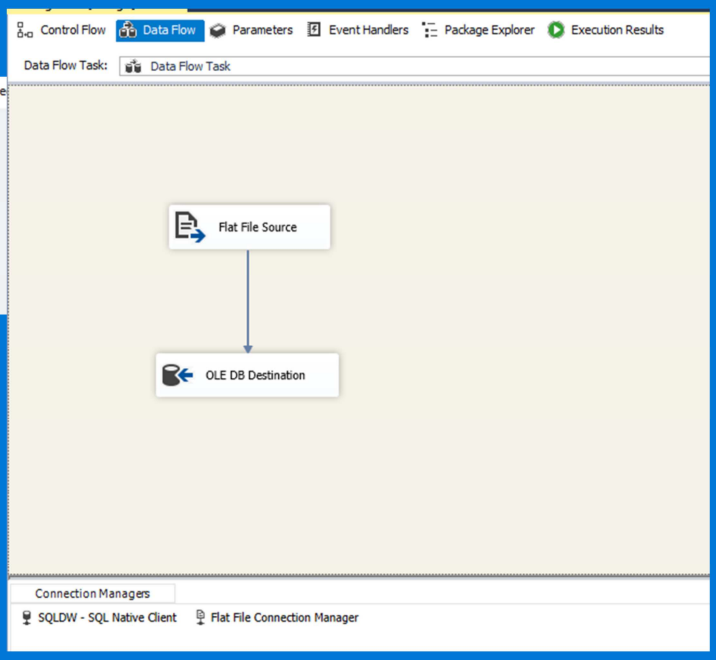
- OLE DB Destination

Connection Manager

- Native SQL Server

SSIS





Demo: Loading Data with SSIS

PolyBase Import

Import data from Hadoop and store it in a MPP DWH table

Use CREATE TABLE AS SELECT to create a new table in the MPP DWH and populate it with the results of a SELECT statement by specifying the name of the external table as the table source in the FROM clause

```
--Import data from Hadoop with CTAS
CREATE TABLE Prod_DB.dbo.TargetTable_EXCH
          WITH (Distribution=HASH (ColA),
                CLUSTERED INDEX (ColB),
                PARTITION (ColC) RANGE RIGHT ON VALUES (xxx, xxx, xxx … ))
AS
SELECT poly.PK_Field, poly.ColA, poly.ColB
        …
FROM   Prod_DB.dbo.ExternalTable_In_Polybase poly
```

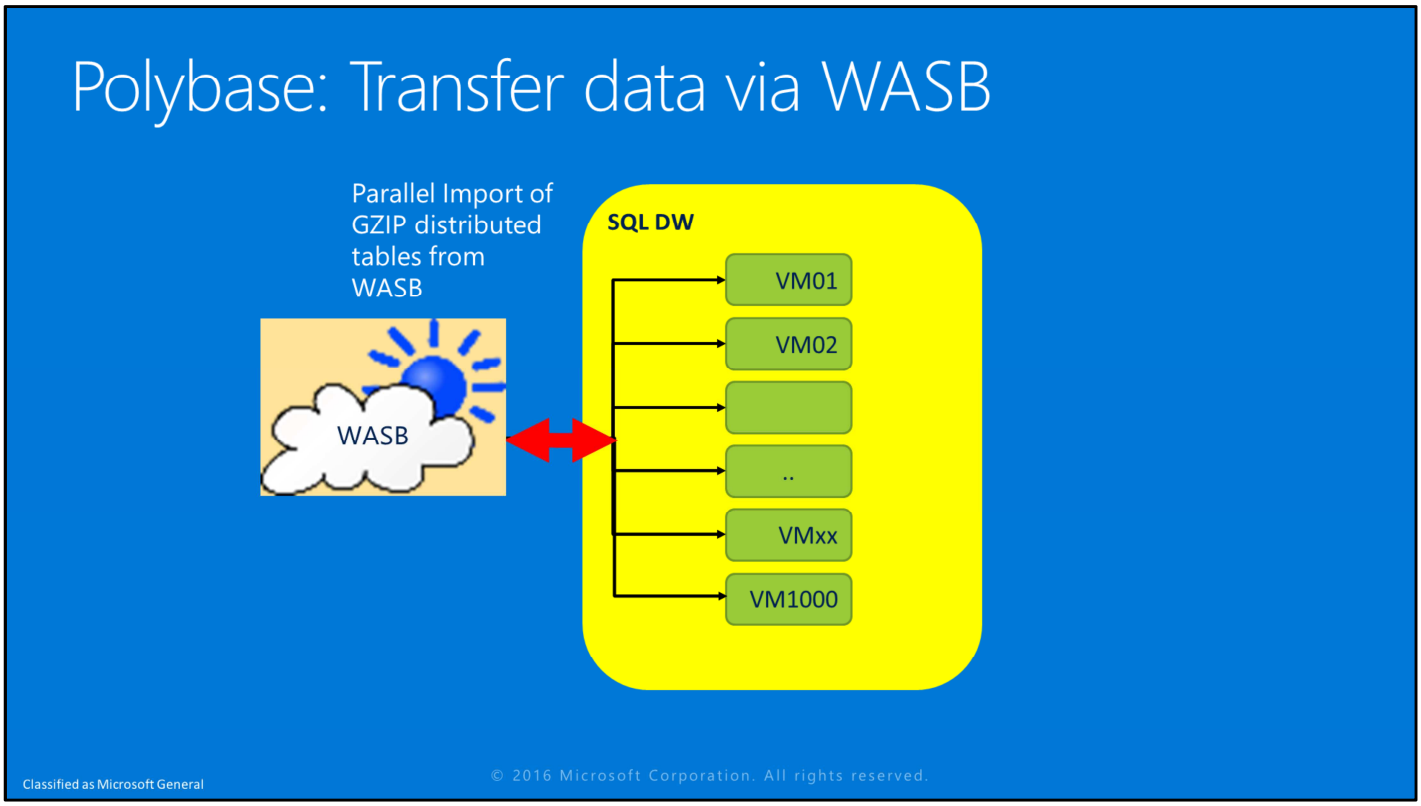

Create Table as Select (CTAS):

- Creates a new table based on a SELECT statement

- Can also be used LOAD DATA from EXTERNAL tables

CTAS:
- Always goes into a new table
  - Allows for minimal logging
  - Can later rename an object to rebind to previous dependencies (e.g., view)
- Can be used to replace SQL's standard INSERT, UPDATE, and DELETE statements
  - Especially with large data operations
- Can transform between replicated and distributed tables

Polybase: Transfer data via WASB



Windows Azure Storage Blob (WASB) is an extension built on top of the HDFS APIs. The WASBS variation uses SSL certificates for improved security. It in many ways "is" HDFS. However, WASB creates a layer of abstraction that enables separation of storage. This separation is what enables your data to persist even when no clusters currently exist and enables multiple clusters plus other applications to access a single piece of data all at the same time. This increases functionality and flexibility while reducing costs and reducing the time from question to insight.

Distributed Data Warehouse Architectures

3rd Party Loading Tools

Informatica PowerCenter (versions up to 9.5.1)

- Windows environments only
- Default operation row by row
- MPP DHW Loader – uses Bulk functionality

SAP Business Objects Data Integrator

Attunity Replicate

- Trickle loading using dwloader under the hood

https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-integrate-solution-partners/

Agenda

- Loading Options
- Loading Tools
- PolyBase
- DMVs
- Exporting Data to Azure Blob Storage

Big Data Integration (recap)

Different types of data

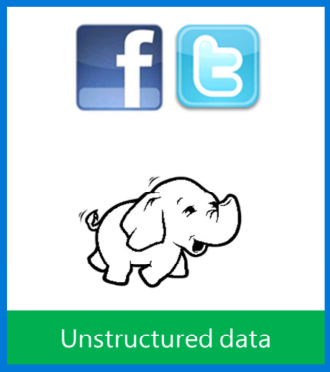
- Webpages, logs, and clicks
- Hardware and software sensors
- Semi-structured/unstructured data

Large scale

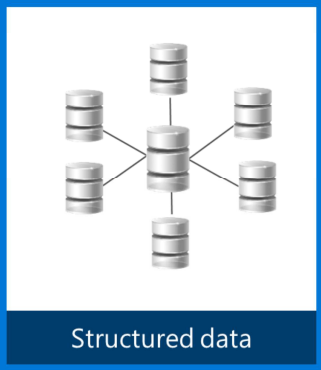
- Hundreds of servers

Advanced data analysis

- Integration between structured and unstructured data
- Power of both



Unstructured data



Structured data

Hadoop (some elements, relevant in this chapter)

HDFS

- Distributed, scalable fault tolerant file system

MapReduce

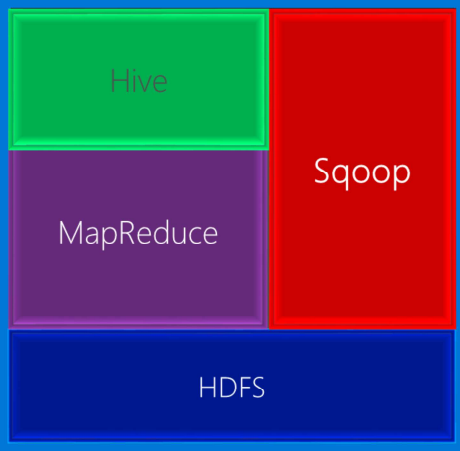
- A framework for writing fault tolerant, scalable distributed applications

Hive

- A relational DBMS that stores its tables in HDFS and uses MapReduce as its target execution language

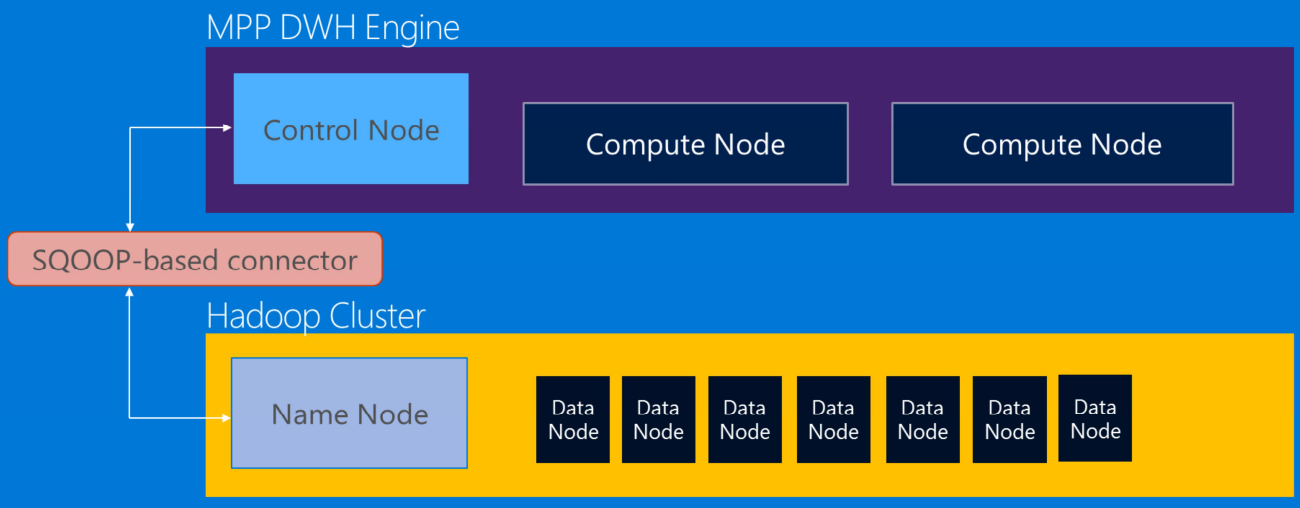
Sqoop

- A library and framework for moving data between HDFS and a relational DBMS



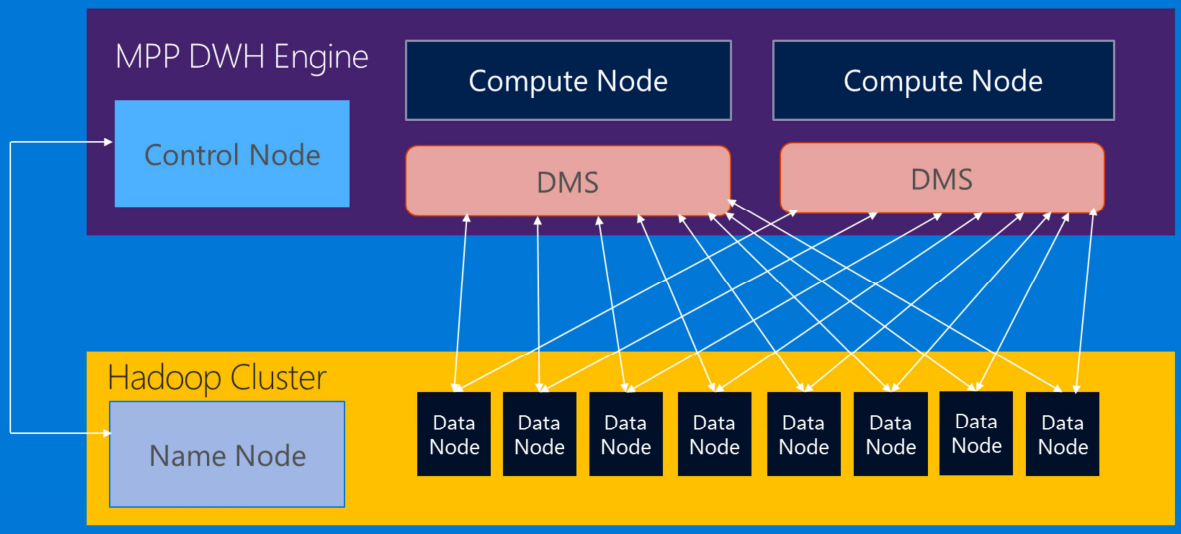
HDInsight

The MPP Engine's Integration Method – **without** PolyBase



HDInsight

The MPP Engine's Integration Method – **with** PolyBase



PolyBase in the Modern Data Warehouse

Background

- Research done by Gray System Lab lead by Technical Fellow David DeWitt

High-level goals for PolyBase

- Seamless Integration with Hadoop via regular T-SQL
- Enhancing the MPP query engine to process data coming from the Hadoop Distributed File System (HDFS)
- Fully parallelized query processing for highly performing data import and export from HDFS
- Integration with various Hadoop implementations
- Hadoop on Windows Server, Hortonworks, and Cloudera

HDFS on WASB[S] (Windows Azure Storage Blobl[s])
https://azure.microsoft.com/en-us/documentation/articles/hdinsight-hadoop-use-blob-storage/

PolyBase Builds The Bridge

Just-in-Time data integration
- Across relational and non-relational data
- High performance parallel architecture
- Fast, simple data loading

Best of both worlds
- Uses computational power at source for both relational data & Hadoop
- Opportunity for new types of analysis

Uses existing analytical skills
- Familiar SQL semantics & behaviour

Query with familiar tools
- SSDT, SQLCMD, PowerBI, …

PolyBase = run time integration

Includes Power BI

Agnostic Architecture

PolyBase is agnostic
=
No vendor lock in

PolyBase supports Hadoop on Linux & Windows

PolyBase integrates with the cloud

PolyBase supports HDInsight

Loosely Coupled Architecture

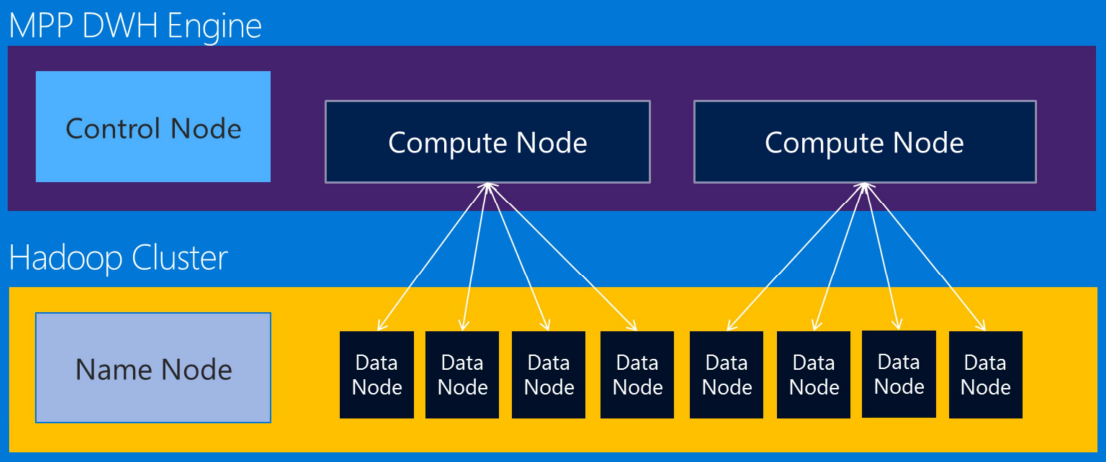
Late Binding Consequences

- Data may change between executions
- Data may change during execution
- Errors identified at run time

All "By Design"
Helps PolyBase keep its agnostic architecture

PolyBase – Bridging the divide…

Parallel data transfers of structured and semi-structured data



PolyBase – Enhancements (since AU3)

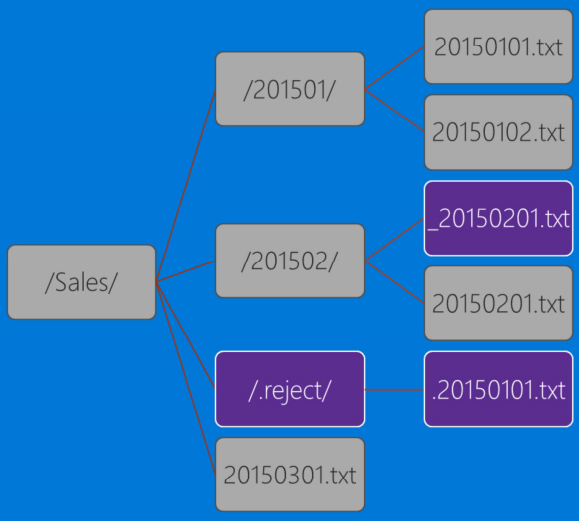
PolyBase: Recursive Directory Traversal

- Enables users to retrieve the content of a folder and all subfolders
- Removes the burden of creating external tables for each subfolder

PolyBase: ORCFile support

- Enables all Polybase scenarios to run against the file format ORCFiles

Recursive Folder Traversal

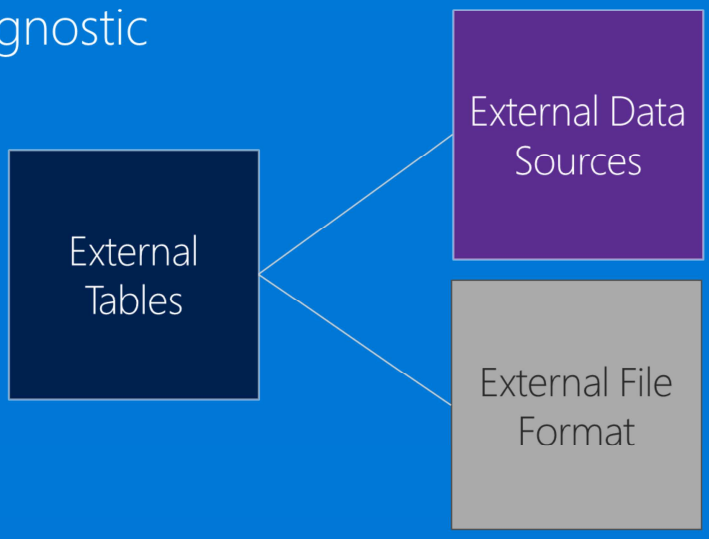


- PolyBase reads data recursively by default
- PolyBase ignores objects (and their children) prefixed by _ or .

Deployment Choices

Table creation - Hadoop

Staying Agnostic



External Tables

- Are metadata, which describe schema of the external data
- Enable data access outside the MPP DWH Database Engine
- They never hold data
- They do not delete data when dropped

Behaviour of an external table, is in the MPP DWH Engine, very similar to Hive external tables

External Table

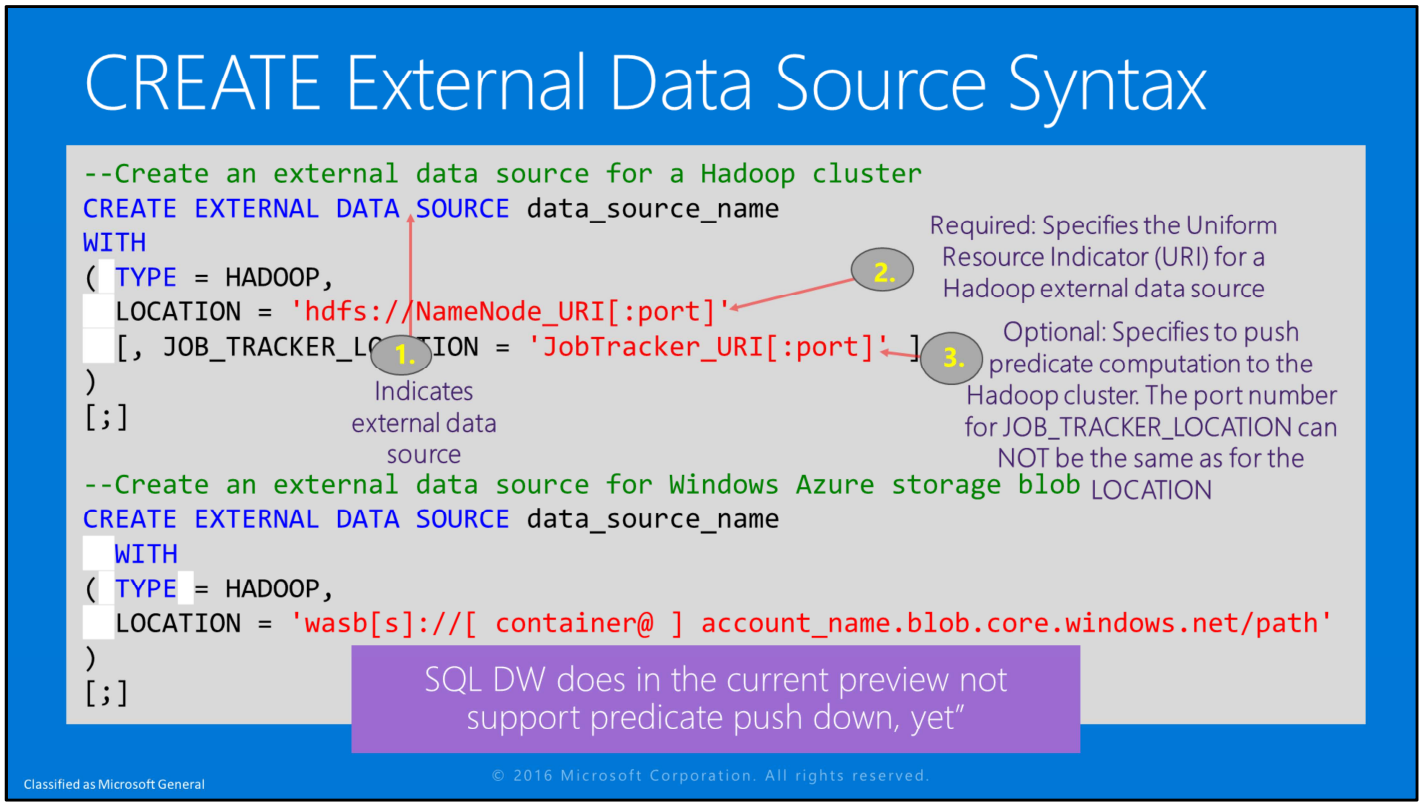
Considerations

- Data can be changed or removed at **any time** on Hadoop side
- PolyBase will **not guarantee** any form of concurrency control or isolation level
- **Same query may return different results**—data gets changed on Hadoop/HDFS side between two query runs
- Query **may fail** if data gets removed or relocated
- **Location of the data** residing on an external cluster gets validated every time a user selects from it

External Tables – Catalog Views

Logical table in shell database (control node)

- sys.external_tables
- sys.tables



Important:

- The LOCATION value is a string and is not validated when you create the external data source. Entering an incorrect value can cause future delays when accessing the location. To avoid typing errors, we recommend that you cut and paste the appropriate example and then replace the italicized parameters with the correct ones for your Hadoop cluster.

- The JOB_TRACKER_LOCATION value is a string and is not validated when you create the external data source. Entering an incorrect value can cause future delays when accessing the location. To avoid typing errors, we recommend that you copy and paste the appropriate example and then replace the italicized parameters with the correct ones for your Hadoop cluster.

CREATE External File Format Syntax

```
--Create an external file format for a Hadoop ORC File.
CREATE EXTERNAL FILE FORMAT file_format_name
WITH ( FORMAT_TYPE = ORC
      [ , DATA_COMPRESSION = {  'org.apache.hadoop.io.compress.SnappyCodec'
                              | 'org.apache.hadoop.io.compress.DefaultCodec'
                             }]) [;]

--Create an external file format for a Hadoop RcFile.
CREATE EXTERNAL FILE FORMAT file_format_name
WITH ( FORMAT_TYPE = RCFILE
      , SERDE_METHOD     = {  'org.apache.hadoop.hive.serde2.columnar.LazyBinaryColumnarSerDe'
                            | 'org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe' }
      [, DATA_COMPRESSION = 'org.apache.hadoop.io.compress.DefaultCodec' ] ) [;]

--Create an external file format for a Hadoop text-delimited file.
CREATE EXTERNAL FILE FORMAT file_format_name
WITH (FORMAT_TYPE = DELIMITEDTEXT
      [, FORMAT_OPTIONS ( [ ,...n ] ) ]
      [, DATA_COMPRESSION = {  'org.apache.hadoop.io.compress.GzipCodec'
                             | 'org.apache.hadoop.io.compress.DefaultCodec' } ] ) [;]

<format_options> ::= {  FIELD_TERMINATOR = field_terminator  | STRING_DELIMITER = string_delimiter
                      | DATE_FORMAT = datetime_format        | USE_TYPE_DEFAULT = { TRUE | FALSE } }
```

1. Indicates external file format

CREATE External Table Syntax

```
--Create a new external table in SQL Server PDW
CREATE EXTERNAL TABLE [ database_name . [ dbo ] . | dbo. ] table_name
    ( <column_definition> [ ,...n ] )
    WITH ( LOCATION = 'hdfs_folder_or_filepath',
           DATA_SOURCE = external_data_source_name,
           FILE_FORMAT = external_file_format_name
           [ , <reject_options> [ ,...n ] ]
    ) [;]

<reject_options> ::=
{
    | REJECT_TYPE = value | percentage
    | REJECT_VALUE = reject_value
    | REJECT_SAMPLE_VALUE = reject_sample_value
}
```

1. Indicates external table
2. Required location of Hadoop file
3. Required Data Source Definition of Hadoop Cluster
4. File Format Definition associated with data import from HDFS (for example, arbitrary field delimiters and reject-related thresholds)

External Table - Sample

```
--STEP 1: Create an external data source for Hadoop
-- DROP EXTERNAL DATA SOURCE FXR_TEST_DSRC;
CREATE EXTERNAL DATA SOURCE FXR_TEST_DSRC
  WITH (  TYPE = HADOOP
        , LOCATION = 'hdfs://192.168.210.145:8020'
        , JOB_TRACKER_LOCATION = '192.168.210.145:8032‘
          ---- defaults:8021 – Cloudera 4.3; 8032 – HDP 2.x on Windows | Cloudera 5.1;
          ----          8050 – HDP 2.x on Linux; 50300 – HDP 1.3
       );
--STEP 2: Create an external file format for a Hadoop text-delimited file.
--DROP EXTERNAL FILE FORMAT FXR_Test_Format;
CREATE EXTERNAL FILE FORMAT FXR_Test_Format
  WITH (  FORMAT_TYPE = DELIMITEDTEXT
        , FORMAT_OPTIONS ( FIELD_TERMINATOR = N';‘
        , USE_TYPE_DEFAULT = TRUE
        , STRING_DELIMITER = '')
       );
```


HDInsight is the same as HDP

HDP =“ Hortonworks HDP”

External Table - Sample (cont.)

```
--STEP 3: Create a new external table in SQL Server MPP SQL
-- DROP EXTERNAL TABLE Test;
CREATE EXTERNAL TABLE Test
    (name nvarchar(17), startzeitpunkt nvarchar(35),
     endzeitpunkt varchar(35), flms_system_realtime nvarchar(19),
     dummy nvarchar(19) NULL, Counter1DTonDur nvarchar(19),
     Counter1DMileage nvarchar(19), dummy2 nvarchar(2) NULL
    )
 WITH
    (LOCATION = '/user/fxr47511/pdwtest'
     , DATA_SOURCE = FXR_TEST_DSRC
     , FILE_FORMAT = FXR_Test_Format
     , REJECT_TYPE = value
     , REJECT_VALUE = 1000
    );
```

External File Format - RCFiles

- RCFile format now supported
- RC = Record Columnar
- Key/value pairs
- Used for storing data in columnar format
- SERDE methods access RCFiles
- Other serde methods can be installed

SERDE
=
Serialization
DeSerialization

External File Format - ORCFiles

- O=Optimised ☺
- Better Compression
- Better Performance
- Requires Hive 0.11 (HDP 2.0) or greater
- Features similar to SQL Server in memory
  - Segment elimination
  - Batch mode execution

External File Format - Limitations

- Row Terminator is fixed as \n
- Encoding is also fixed : UTF8
- Compression choice may be limited by format

External File Format Notes

LazyBinaryColumnarSerDe is significantly faster and more efficient than ColumnarSerDe

ORC is the direction of travel for data in Hive

Data Compression not designed for the Hadoop Region as the IB connectivity is so fast

Data Compression more beneficial for external clusters using low speed networks

Balanced Execution – Conclusion

Is essentially a divide and conquer challenge

- Break the task up into small enough pieces
- Spread those pieces round as evenly as we can

How do we do this for Hadoop?

Table Level Statistics

When an external table is created, table level statistics are also persisted

- Row count
- Page count

https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-tables-statistics/

Parallel Transfer Concepts

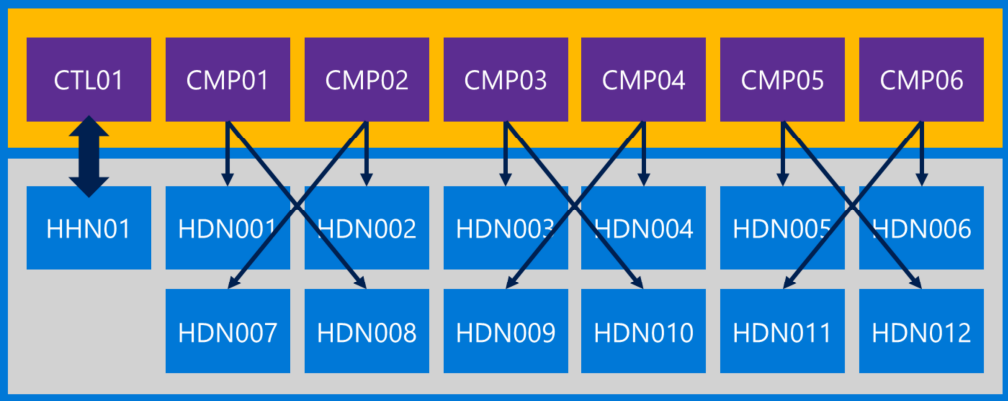
Maximise Throughput

- Every compute node in the MPP Engine sees every data node in Hadoop
- Ensure direct connections are established between all scale out nodes of the MPP Engine & Hadoop

Balanced Execution

- Ensure all nodes are equally busy when reading and writing data

Maximising Throughput



Agenda

- Loading Options
- Loading Tools
- PolyBase
- DMVs
- Exporting to Azure Blob Storage

Loader DMVs

- Run_ID
- Request_ID is a GUID as opposed to QID

DMVs

- sys.pdw_loader_run_stages
- sys.pdw_loader_backup_runs
- sys.pdw_loader_backup_run_details

https://azure.microsoft.com/en-us/documentation/articles/sql-data-warehouse-reference-tsql-system-views/

DMV Limits on History

sys.dm_pdw_exec_sessions

- 10,000 most recent sessions

sys.dm_pdw_exec_requests

- 10,000 most recent requests

sys.dm_pdw_errors

- 10,000 most recent errors

sys.dm_pdw_sql_requests

- 1,000 most recent SQL requests

sys.dm_pdw_request_steps

- All steps for 1,000 most recent SQL requests

sys.dm_pdw_dms_workers

- All workers for 1,000 most recent SQL requests

System View	Maximum Rows
sys.dm_pdw_component_health_alerts	10,000
sys.dm_pdw_dms_cores	100
sys.dm_pdw_dms_datachannel_credit	10,000
sys.dm_pdw_dms_datachannel_threads	10,000
sys.dm_pdw_dms_messages	10,000
sys.dm_pdw_dms_workers	Total number of DMS workers for the most recent 1000 SQL requests.
sys.dm_pdw_dms_worker_pairs	10,000
sys.dm_pdw_errors	10,000
sys.dm_pdw_exec_requests	10,000
sys.dm_pdw_exec_sessions	10,000
sys.dm_pdw_recent_activity	1,000
sys.dm_pdw_os_event_logs	10,000
sys.dm_pdw_sql_requests	The most recent 1000 SQL requests that are stored in sys.dm_pdw_exec_requests.
sys.dm_pdw_request_steps	Total number of steps for the most recent 1000 SQL requests that are stored in sys.dm_pdw_exec_requests.

Capturing History

- DMV data is transient; thresholds easily exceeded by load
- Need to retain the history ourselves
- Labels are a good start
- More information is generally required
- Persisting data in user tables is popular

Consider "stamping" stored procedures with a dummy parameter value (e.g. SSIS execution GUID).
This value will appear in command text of sys.dm_pdw_exec_requests
By scanning command text for the dummy value all queries associated to the session can be identified

Capturing History

```
CREATE VIEW [dbo].[vSessionRequestMetaData]
AS
SELECT
 s.Session_ID       as Session_ID                       ,[label]       as Request_QueryLabel
,s.[status]         as Session_Status                   ,command       as Request_Command
,s.login_name       as Session_LoginName                ,database_id   as Request_Database_ID
,s.login_time       as Session_LoginTime                ,e.source      as Error_Source
,r.request_id       as Request_ID                       ,e.[type]      as Error_Type
,r.[status]         as Request_Status                   ,e.create_time as Error_CreateTime
,submit_time        as Request_SubmitTime               ,e.pdw_node_id as Error_PDWNodeID
,start_time         as Request_StartTime                ,e.spid        as Error_SPID
,end_compile_time   as Request_EndCompileTime          ,e.thread_id    as Error_Thread
,end_time           as Request_EndTime                  ,e.details     as Error_Details
,total_elapsed_time as Request_TotalElapsedDuration_ms ,SYSDATETIME()  as Log_Date
,DATEDIFF(ms,submit_time,start_time)     as Request_InitiateDuration_ms
,DATEDIFF(ms,start_time,end_compile_time)as Request_CompileDuration_ms
,DATEDIFF(ms,end_compile_time,end_time)  as Request_ExecDuration_ms
FROM sys.dm_pdw_exec_requests r
JOIN sys.dm_pdw_exec_sessions s  ON  r.session_id = s.session_id
LEFT JOIN sys.dm_pdw_errors e    ON  r.error_id   = e.error_id
                                 AND r.session_id = e.session_id
                                 AND r.request_id = e.request_id;
```

Capturing History

```
CREATE PROC [dbo].[csp_ETL_LogState]
@pRunGUID NVARCHAR(38),@pPkgName NVARCHAR(256),@pLogGUID NVARCHAR(38)
AS
INSERT INTO dbo.ETL_ProcessLog
SELECT  @pRunGUID as RunGUID
,       @pPkgName as PkgName
,       @pLogGUID as LogGUID
,       m.*
,       SYSDATETIME() AS Log_Date
FROM dbo.vSessionRequestMetadata m
JOIN (Select Session_ID
from dbo.vSessionRequestMetadata where Request_Command like '%'+@pLogGUID+'%'
AND Session_ID <> Session_ID()
GROUP BY Session_ID
) s
ON m.Session_ID = s.Session_ID
OPTION (Label='dbo.csp_ETL_LogState : Step 1 : Insert into ETL_ProcessLog')
```

Field Note on Logging Tables

Tend to be

- Quite Small
- Write Heavy

When Replicated

- Slows ELT process
- Can cause deadlocks

Top Tip
Distribute the table for enhanced write speed

Agenda

- Loading Options
- Loading Tools
- PolyBase
- DMVs
- Exporting to Azure Blob Storage