Implementing a Cloud Data Warehouse in Microsoft Azure Synapse Analytics

UNDERSTANDING MICROSOFT AZURE SYNAPSE ANALYTICS



Gary Grudzinskas
AZURE CONSULTANT AND AUTHOR
@garygrudzinskas



Overview



What is Azure Synapse Analytics (formally SQL Data Warehouse)

When to use Azure Synapse Analytics

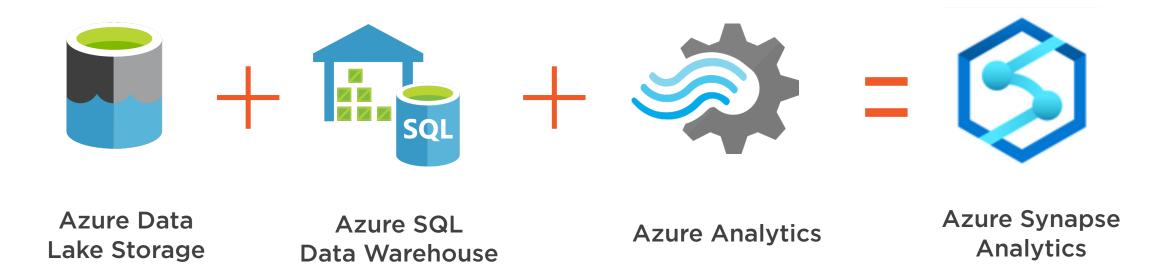
Massive Parallel Processing

Data distributions in Azure Synapse Analytics

Partitioning data in Azure Synapse Analytics

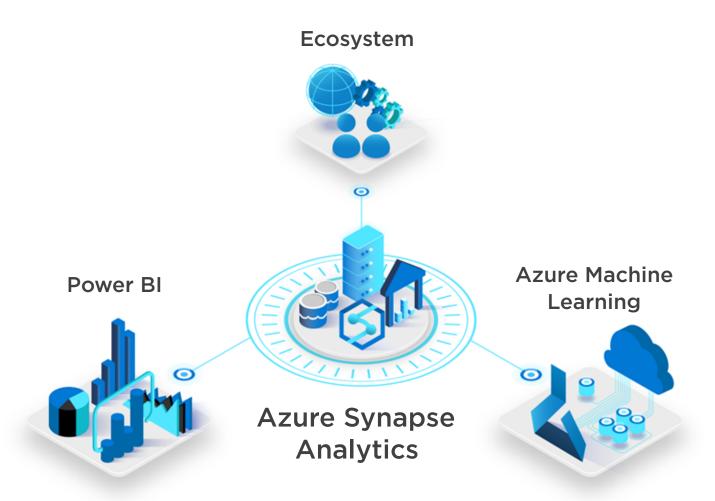


Understanding Azure Synapse Analytics





Understanding Azure Synapse Analytics





Understanding Azure Synapse Analytics















Azure SQL Data Warehouse

A cloud-based enterprise data warehouse (EDW) that uses massively parallel processing (MPP) to run complex queries across petabytes of data quickly.



Knowing When to Use Azure Synapse Analytics



SQL Data Warehouse is the most appropriate solution when you need to keep historical data separate from source transaction systems for performance reasons.



How Organizations Can Use SQL Data Warehouse



Independently size compute power, regardless of storage needs.



Grow or shrink compute power without moving data.



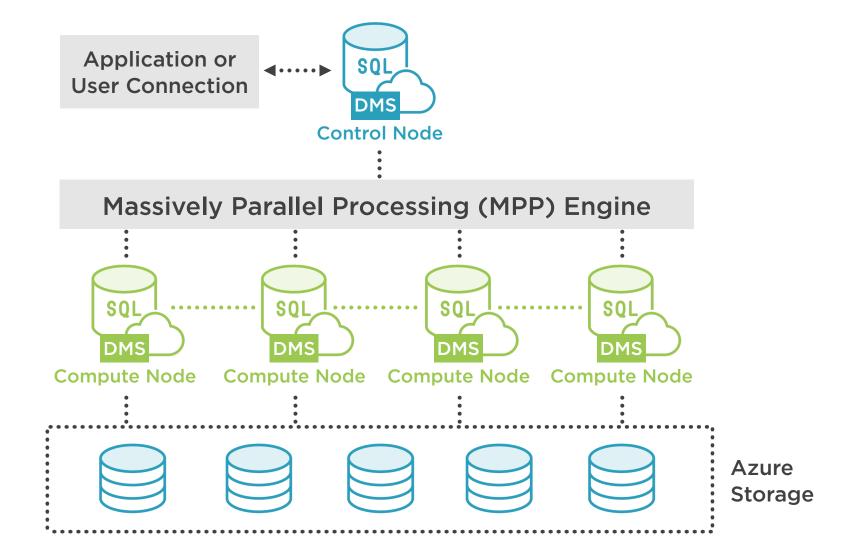
Pause compute capacity while leaving data intact, so you pay only for storage.



Resume compute capacity during operational hours.



Understanding Massive Parallel Processing





Control Node

The front end that interacts with all applications and connections. The MPP engine runs on the control node to optimize and coordinate parallel queries.



Compute Node

Provide the computational power for analytics. Separated from storage nodes. These are scaled using data warehouse units (DWU)



Data Warehouse Unit (DWU)

A collection of analytic resources that are previsioned. This is a combination of CPU, memory, and IO. These can be scaled to up or down to meet needs.



Data Movement Service (DMS)

Data transport technology that coordinates data movement between compute nodes. When SQL Data Warehouse runs a query, the work is divided into 60 smaller queries that run in parallel.



Storage Node

Separate from Compute in order to keep data at rest. This is cheaper than data that is being analyzed.



Implementing Data Distribution for an SQL Data Warehouse



A distribution is the basic unit of storage and processing for parallel queries

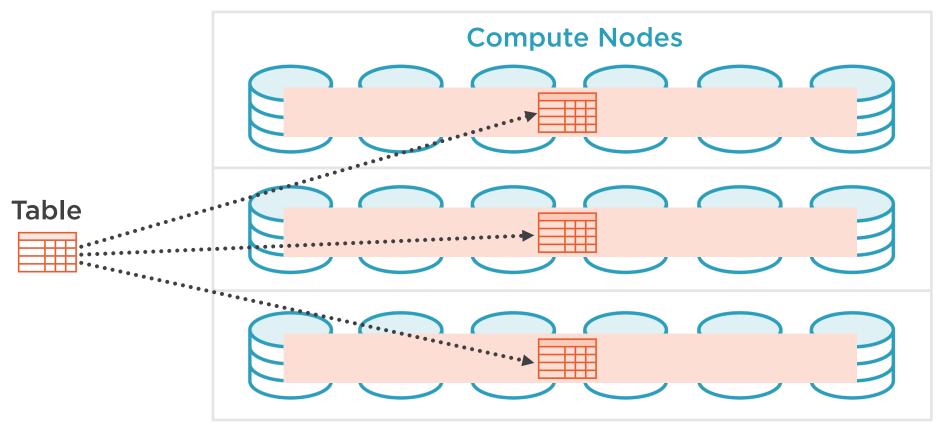
Rows are stored across 60 distributions which are run in parallel

Each Compute node manages one or more of the 60 distributions



Replicated Tables

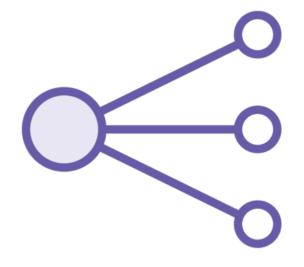
Caches a full copy on each compute node. Used for small tables.







Round Robin

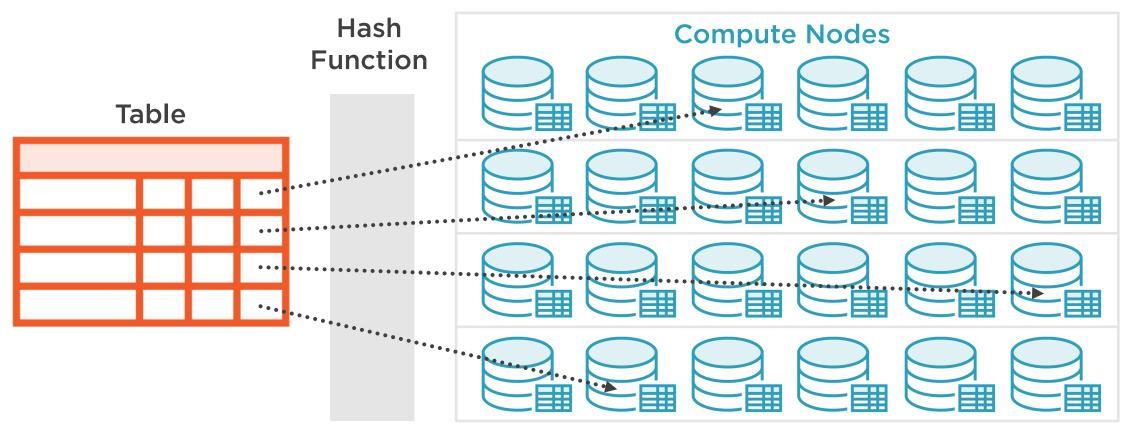


Distributes data evenly across the table without additional optimization



Hash-Distributed Table

A hash function is used to assign each row to one distribution deterministically



What Data Distribution to Use?

Туре	Great fit for	Watch out if
Replicated	Small-dimension tables in a star schema with less than 2GB of storage after compression (~5x compression)	Many write transactions are on the table (insert/update/delete)
		You change DWU provisioning frequently
		You use only 2-3 columns, but your table has many columns
		You index a replicated table
Round-robin (default)	Temporary/Staging table	Performance is slow due to data movement
	No obvious joining key or good candidate column.	
Hash	Fact tables	The distribution key can't be updated.
	Large-dimension tables	





Implementing Partitions for an SQL Data Warehouse



Table partitions enable you to divide your data into smaller groups of data



Improve the efficiency and performance of loading data by use of partition deletion, switching and merging



Usually data is partitioned on a date column tied to when the data is loaded into the database



Can also be used to improve query performance



Table Partitions

Clustered Columnstore

Clustered Index

Heap

Updateable primary storage method.

Great for read-only.

An index that is physically stored in the same order as the data being indexed.

Data is not in any particular order.

Use when data has no natural order.



Sizing Partitions

Creating a table with too many partitions can hurt performance under some circumstances.

Usually a successful partitioning scheme has 10 or a few hundred partitions.

Clustered columnstore tables, it is important to consider how many rows belong to each partition.

Before partitions are created, SQL Data Warehouse already divides each table into 60 distributed databases.

