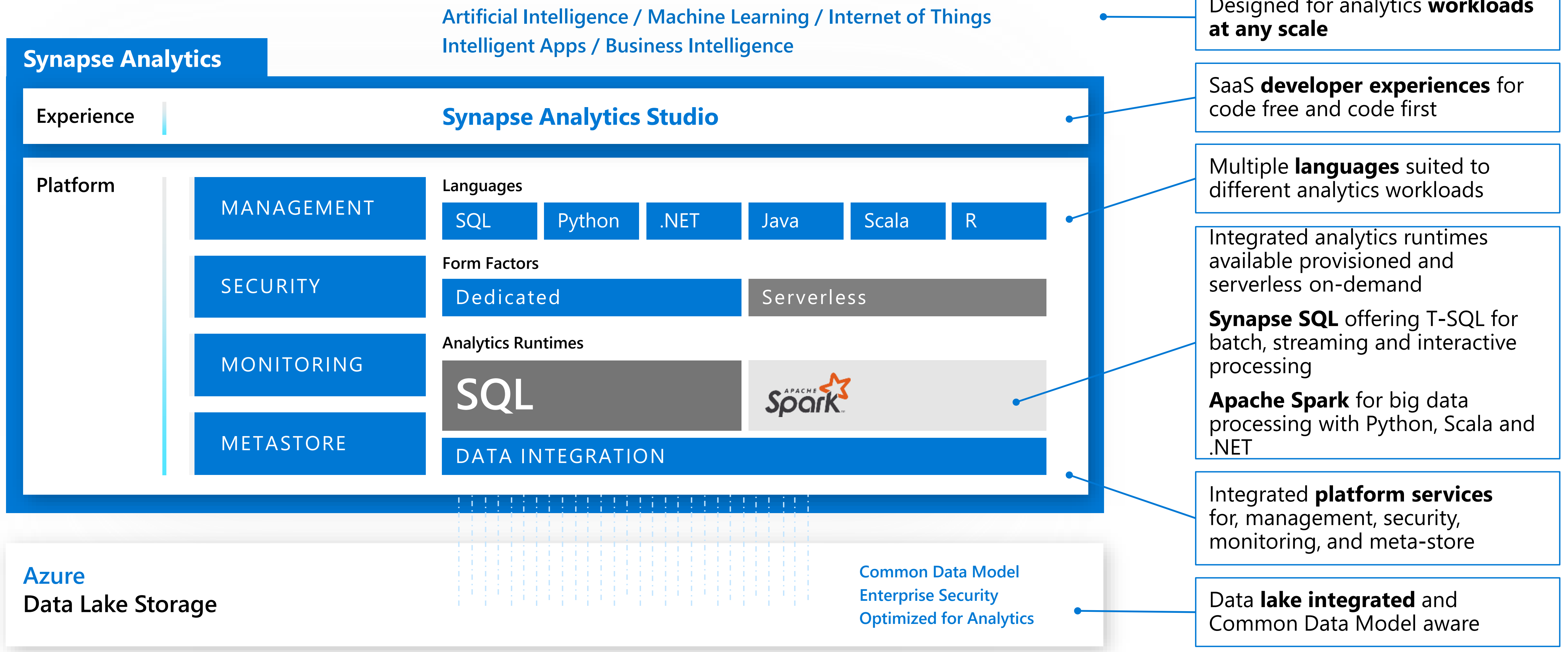
Aas Trailblazers

Unleashing insights

Azure Synapse Analytics

Limitless analytics service with unmatched time to insight



Designed for analytics **workloads at any scale**

SaaS **developer experiences** for code free and code first

Multiple **languages** suited to different analytics workloads

Integrated analytics runtimes available provisioned and serverless on-demand

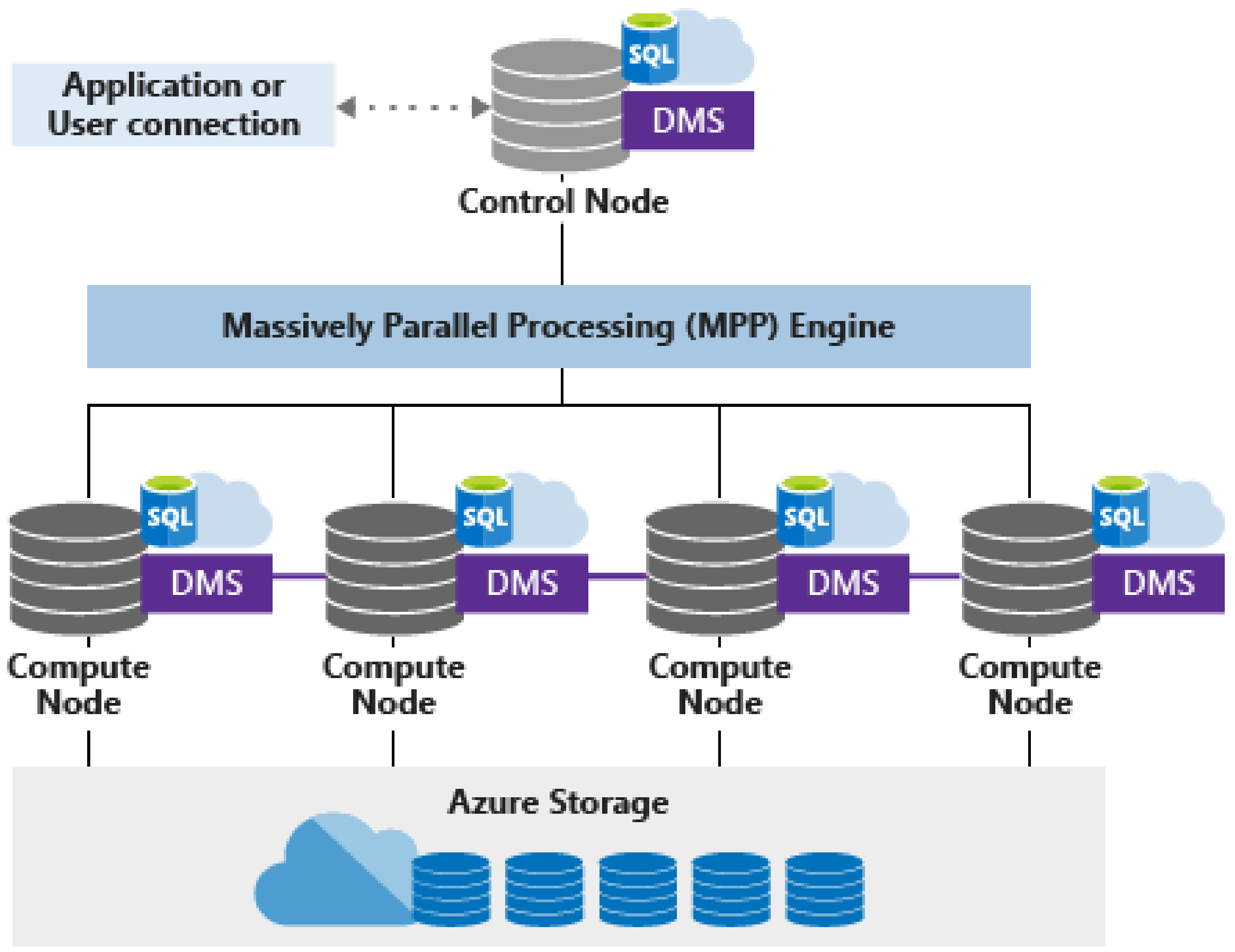
Synapse SQL offering T-SQL for batch, streaming and interactive processing

Apache Spark for big data processing with Python, Scala and .NET

Integrated **platform services** for, management, security, monitoring, and meta-store

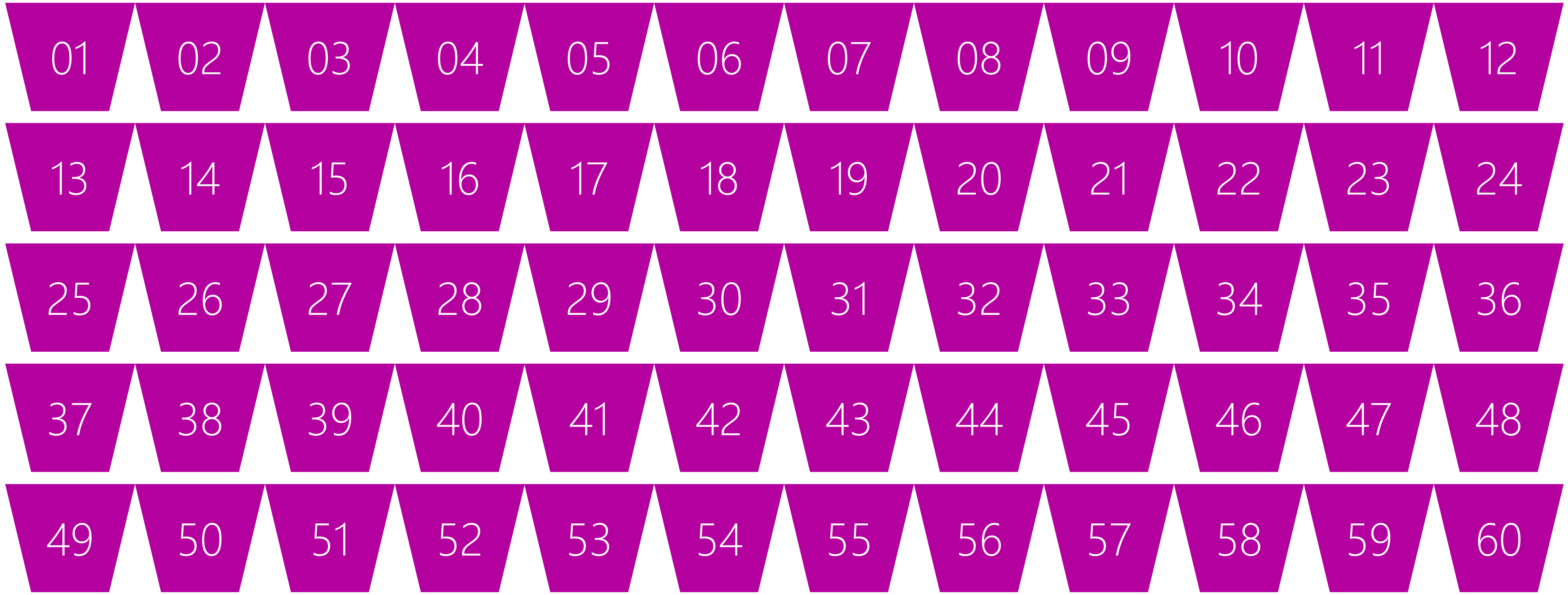
Data **lake integrated** and Common Data Model aware

Synapse SQL – MPP Architecture



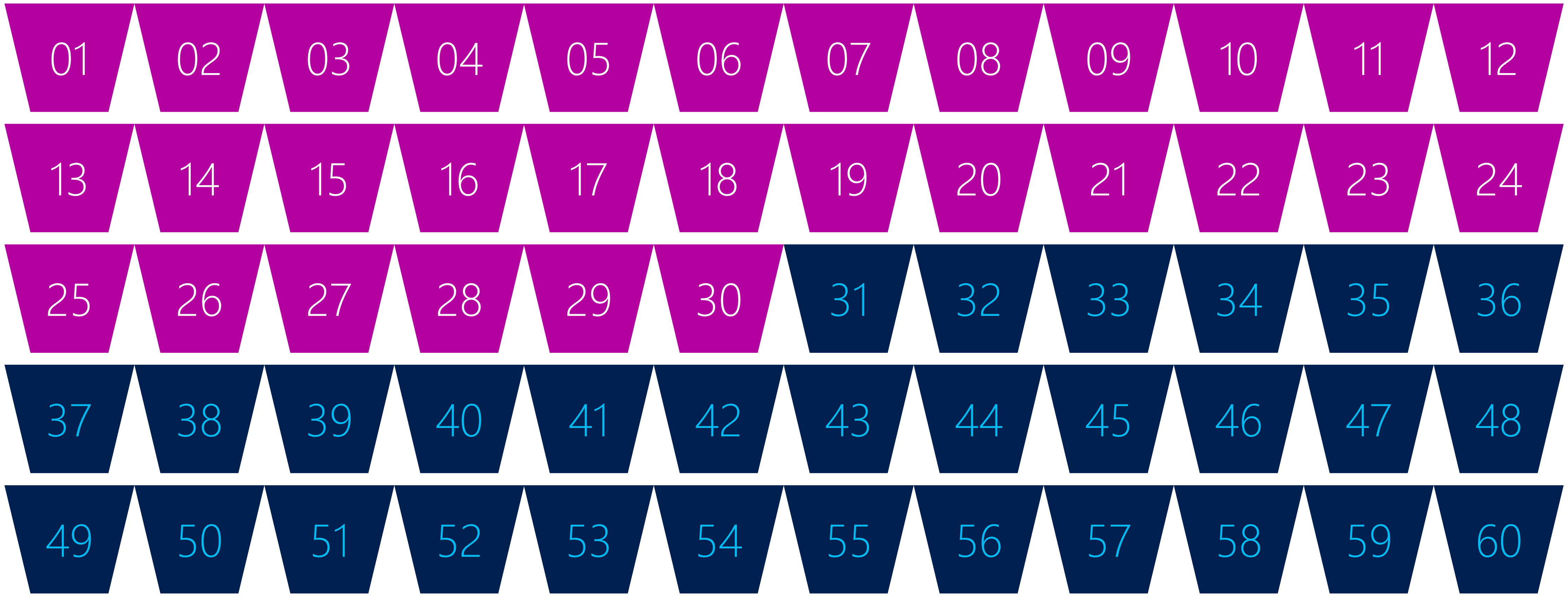
Mapping Compute in dedicated SQL Pool

1 Compute Node



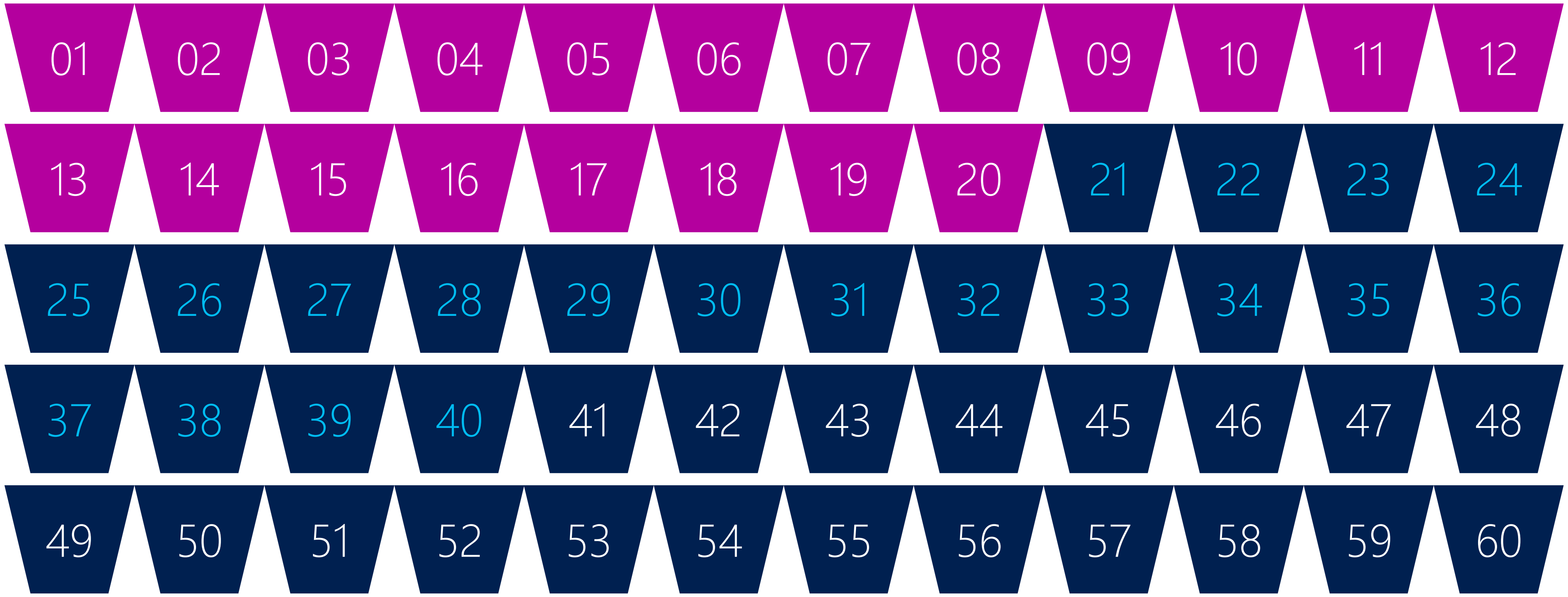
Mapping Compute in dedicated SQL Pool

2 Compute Nodes



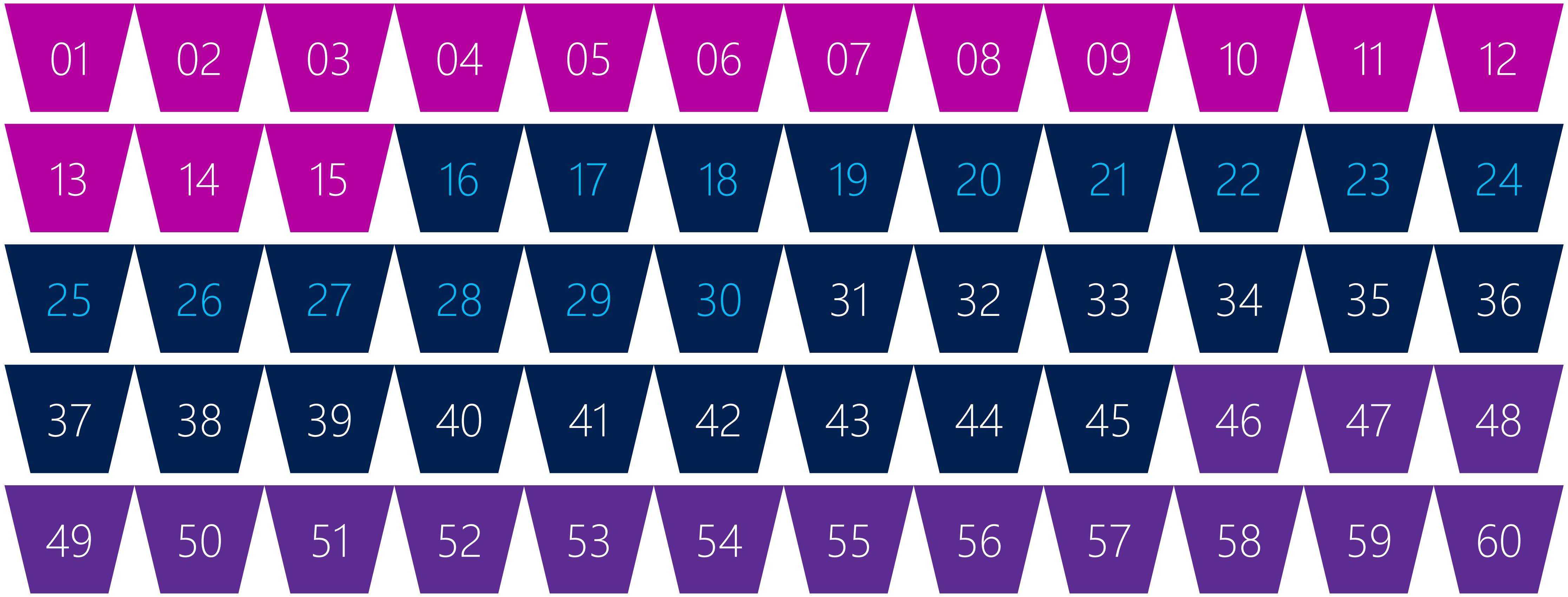
Mapping Compute in dedicated SQL Pool

3 Compute Nodes



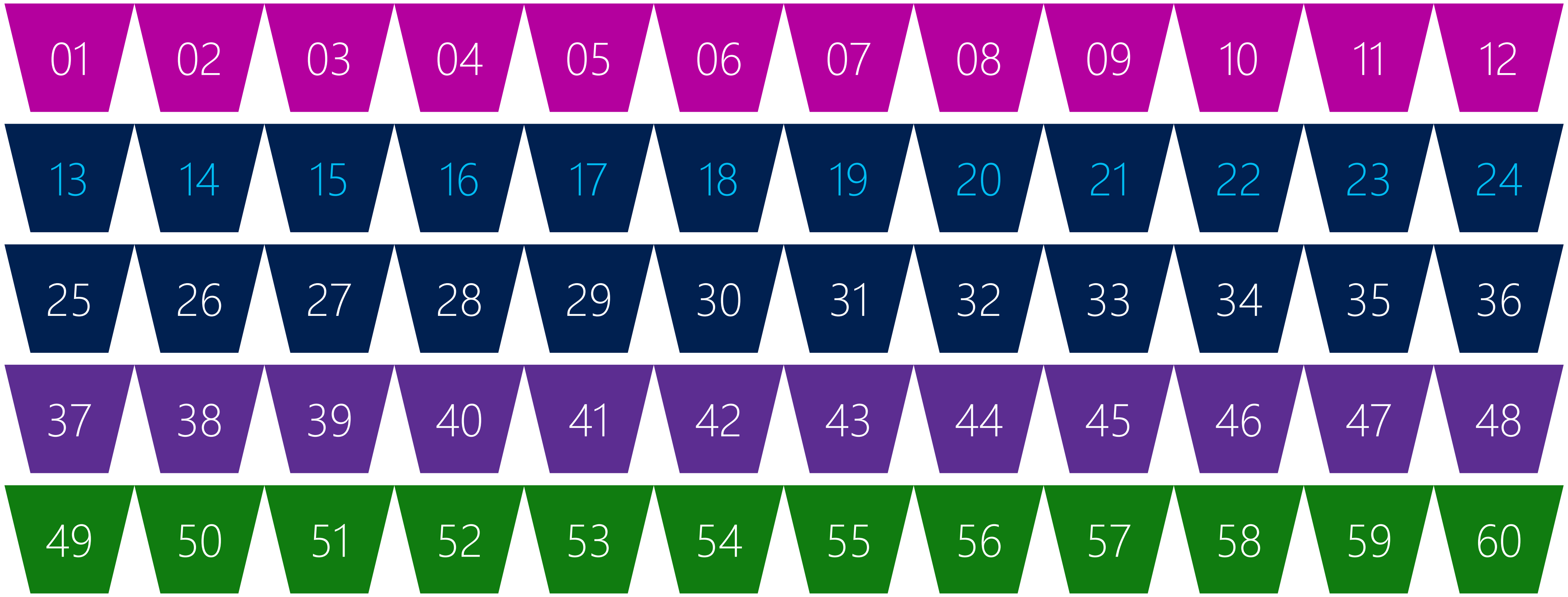
Mapping Compute in dedicated SQL Pool

4 Compute Nodes



Mapping Compute in dedicated SQL Pool

5 Compute Nodes



Pausing compute in dedicated SQL Pool

Compute Nodes - Released

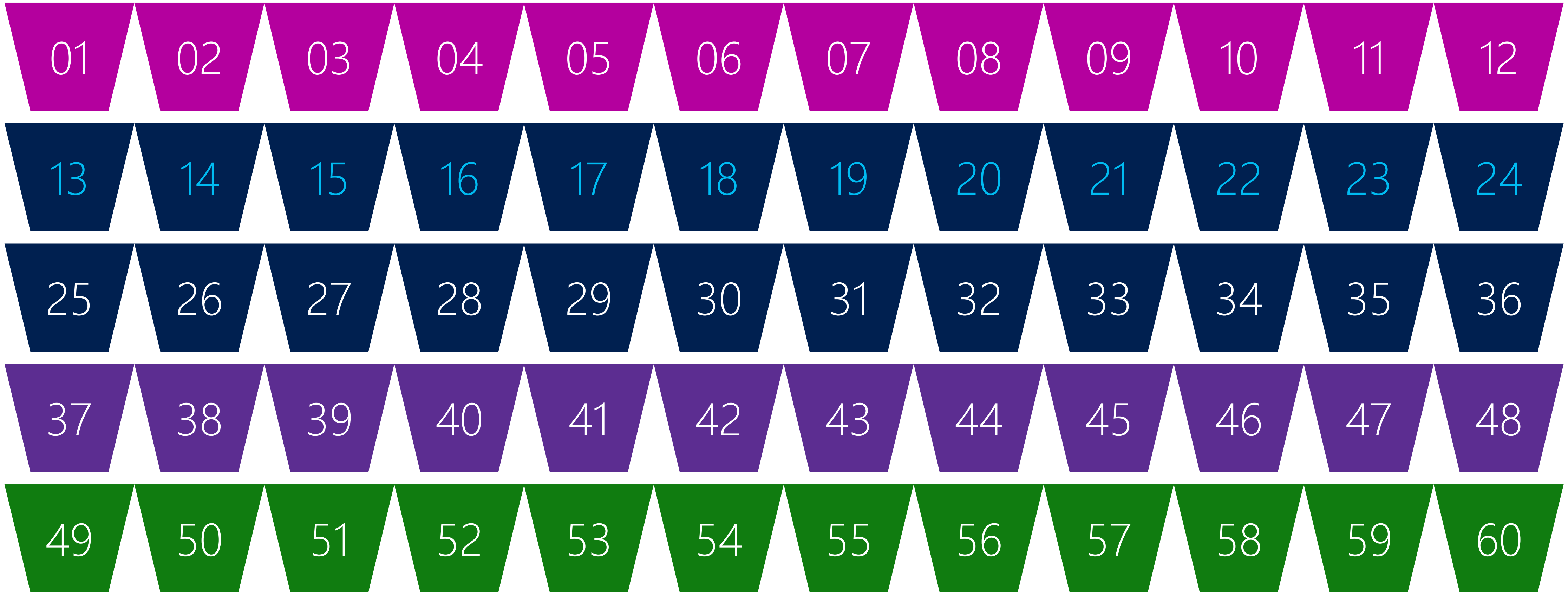
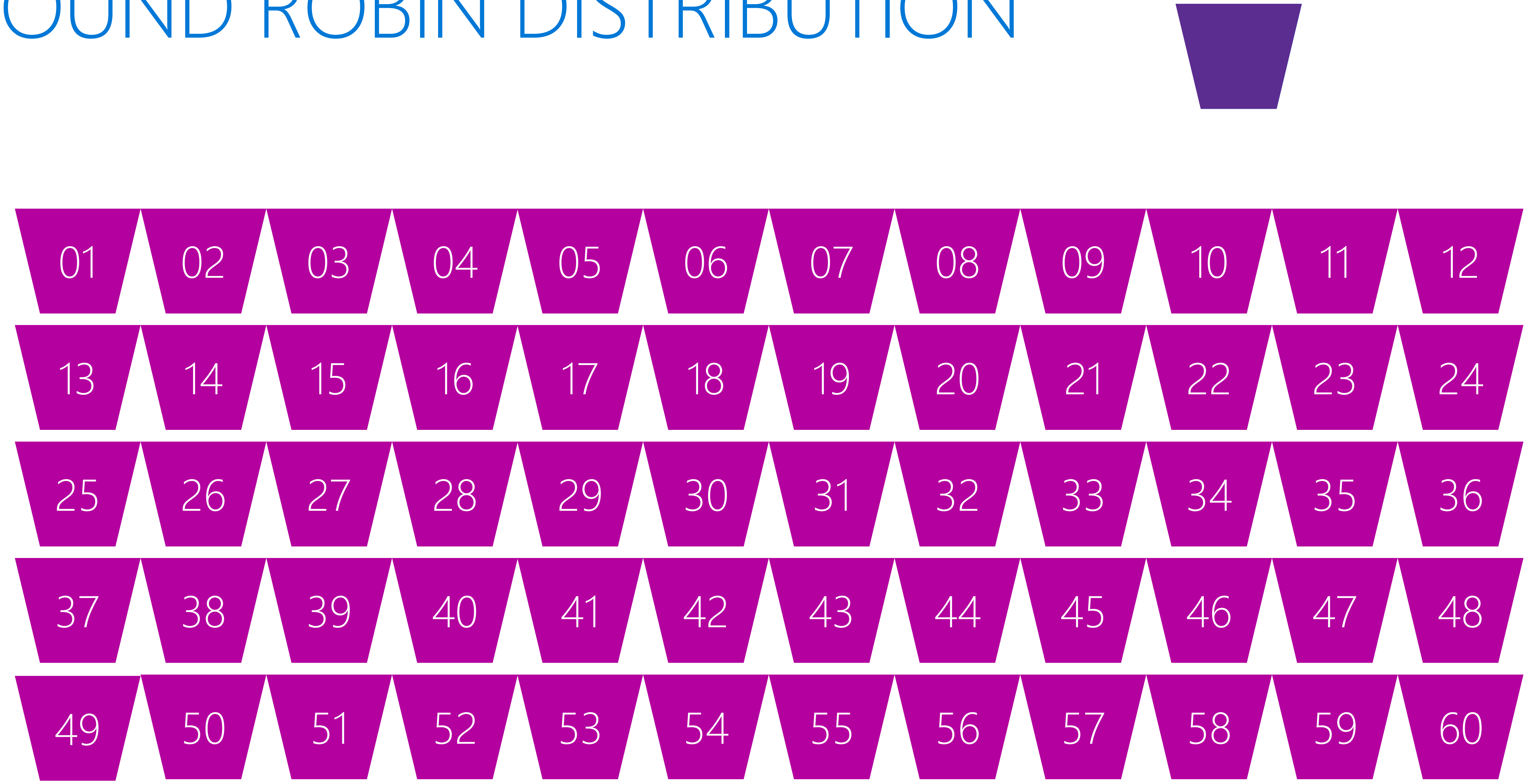


Table types

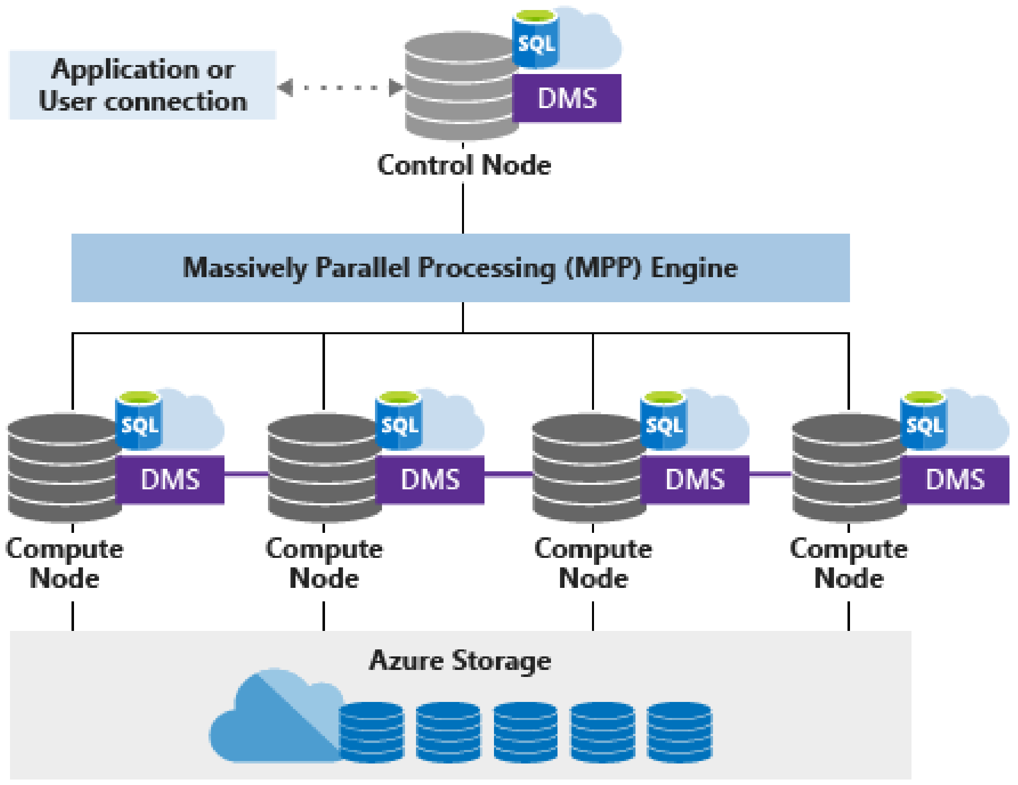
- Distributed table
  - **Round-Robin (default)**
  - **Hash key**
- Replicated table
- External table

ROUND ROBIN DISTRIBUTION



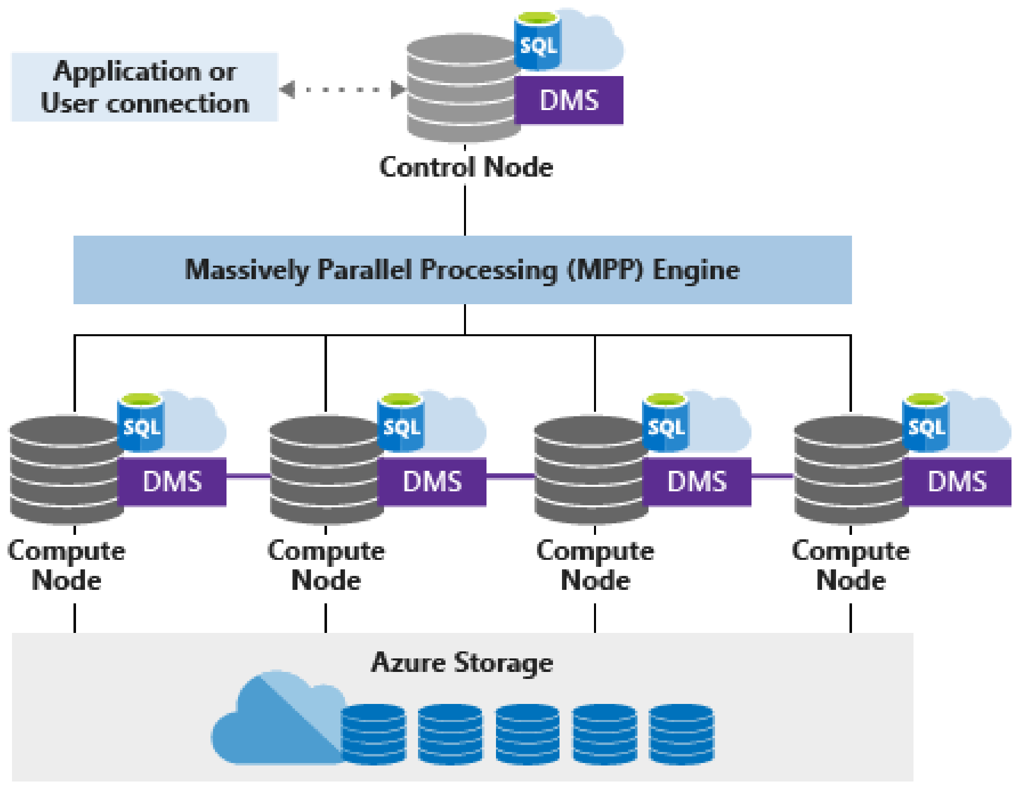
Synapse SQL – MPP Architecture

111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)



Synapse SQL – MPP Architecture

111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)

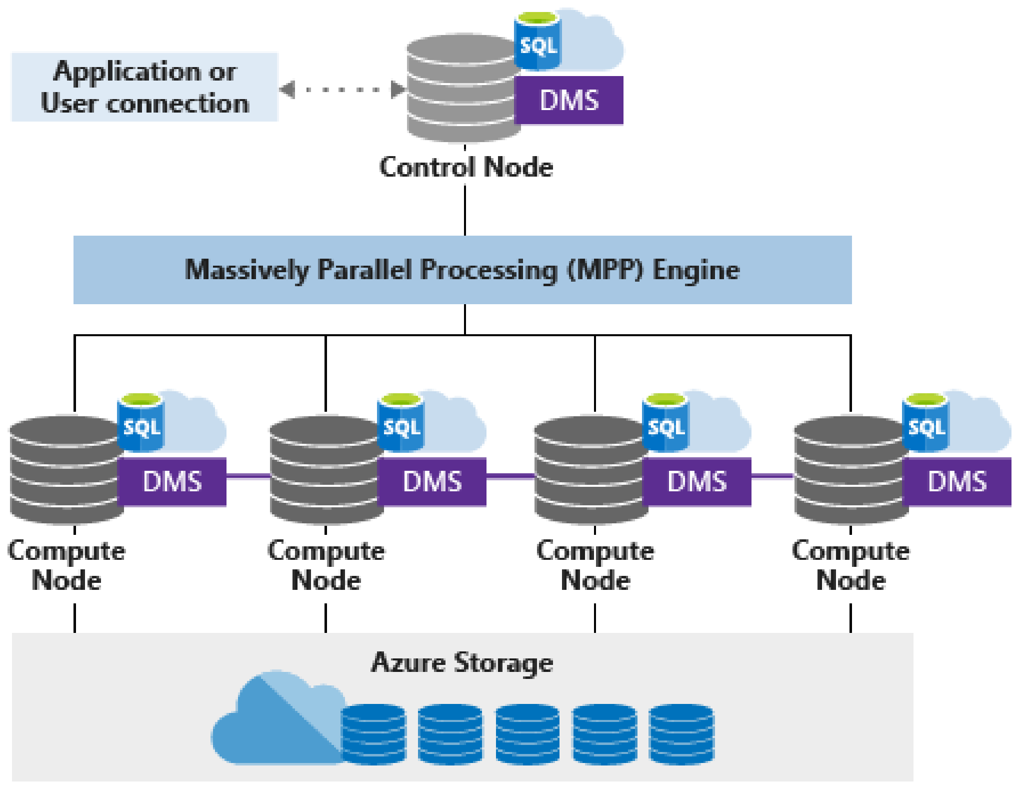


Round Robin Distribution

111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)		111 (Amt, xxx, xxx)

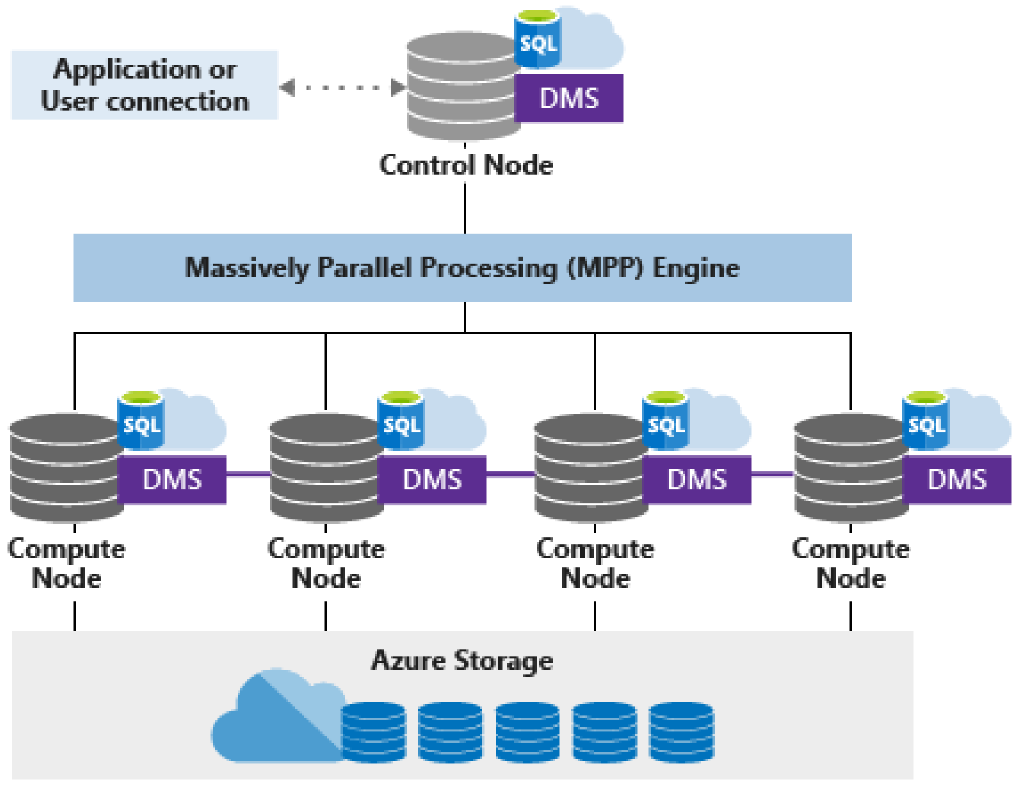
Synapse SQL – MPP Architecture

```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```



111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)		111 (Amt, xxx, xxx)

Synapse SQL – MPP Architecture



```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```

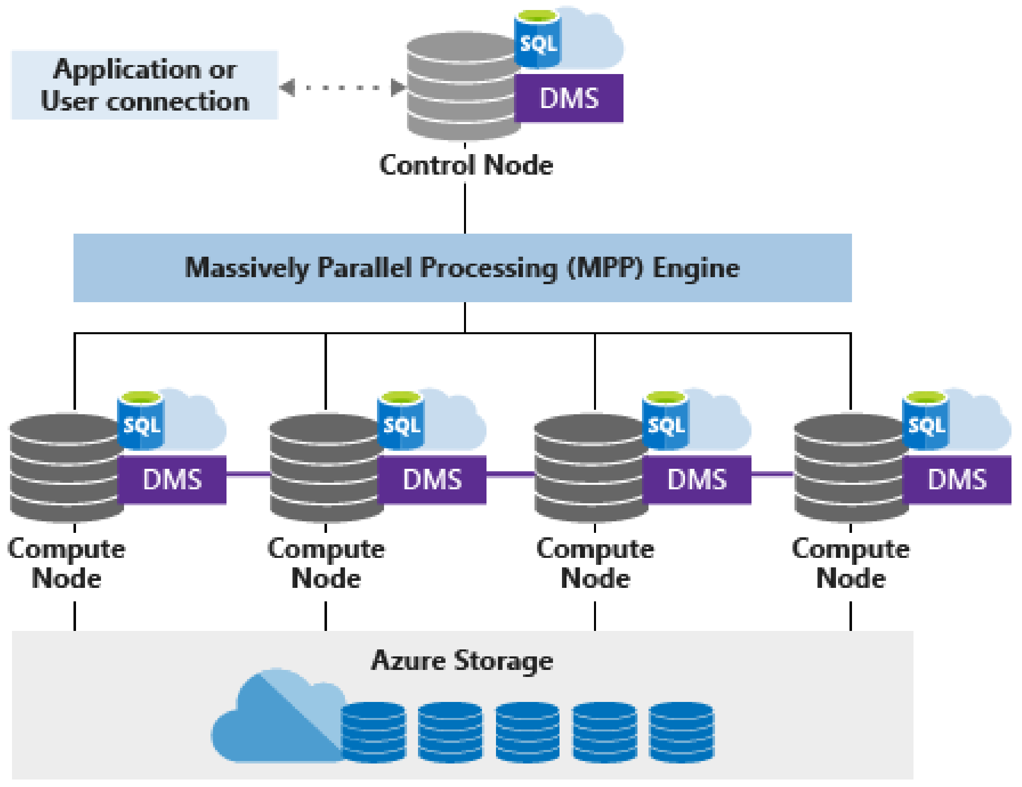
222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)	333 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)	
222 (Amt, xxx, xxx)		444 (Amt, xxx, xxx)	
		444 (Amt, xxx, xxx)	
		444 (Amt, xxx, xxx)	

Shuffle data across based on ProductKey to serve query

https://azure.microsoft.com/en-us/blog/lightning-fast-query-performance-with-azure-sql-data-warehouse/

Synapse SQL – MPP Architecture

```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```



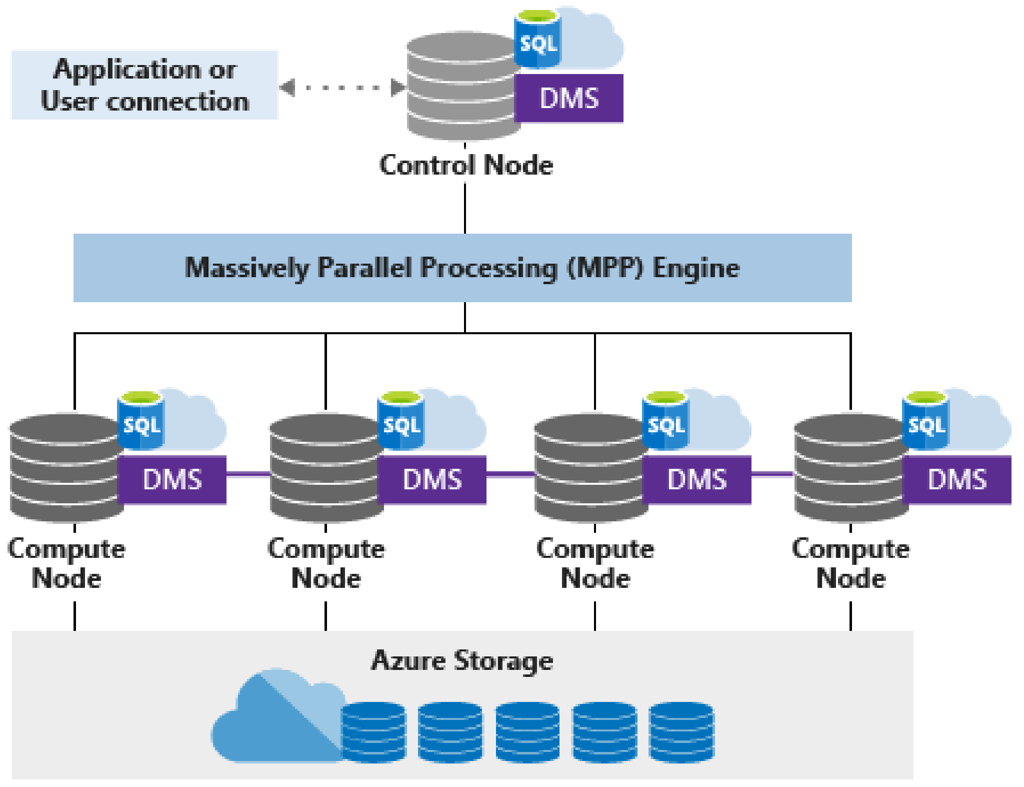
222 (Sum(Amt)) 111 (Sum(Amt)) 555 (Sum(Amt)) 333 (Sum(Amt))

444 (Sum(Amt))

Synapse SQL – MPP Architecture

111 (Sum(Amt))
222 (Sum(Amt))
333 (Sum(Amt))
444 (Sum(Amt))
555 (Sum(Amt))

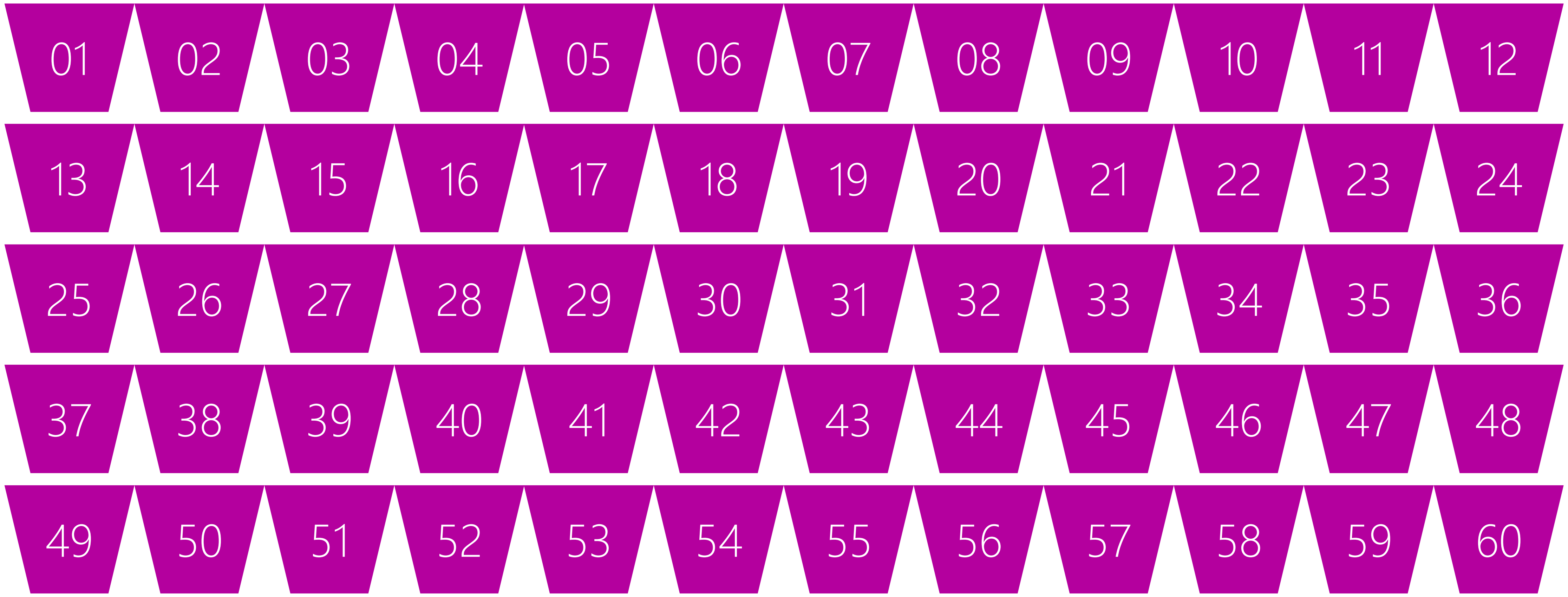
```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```



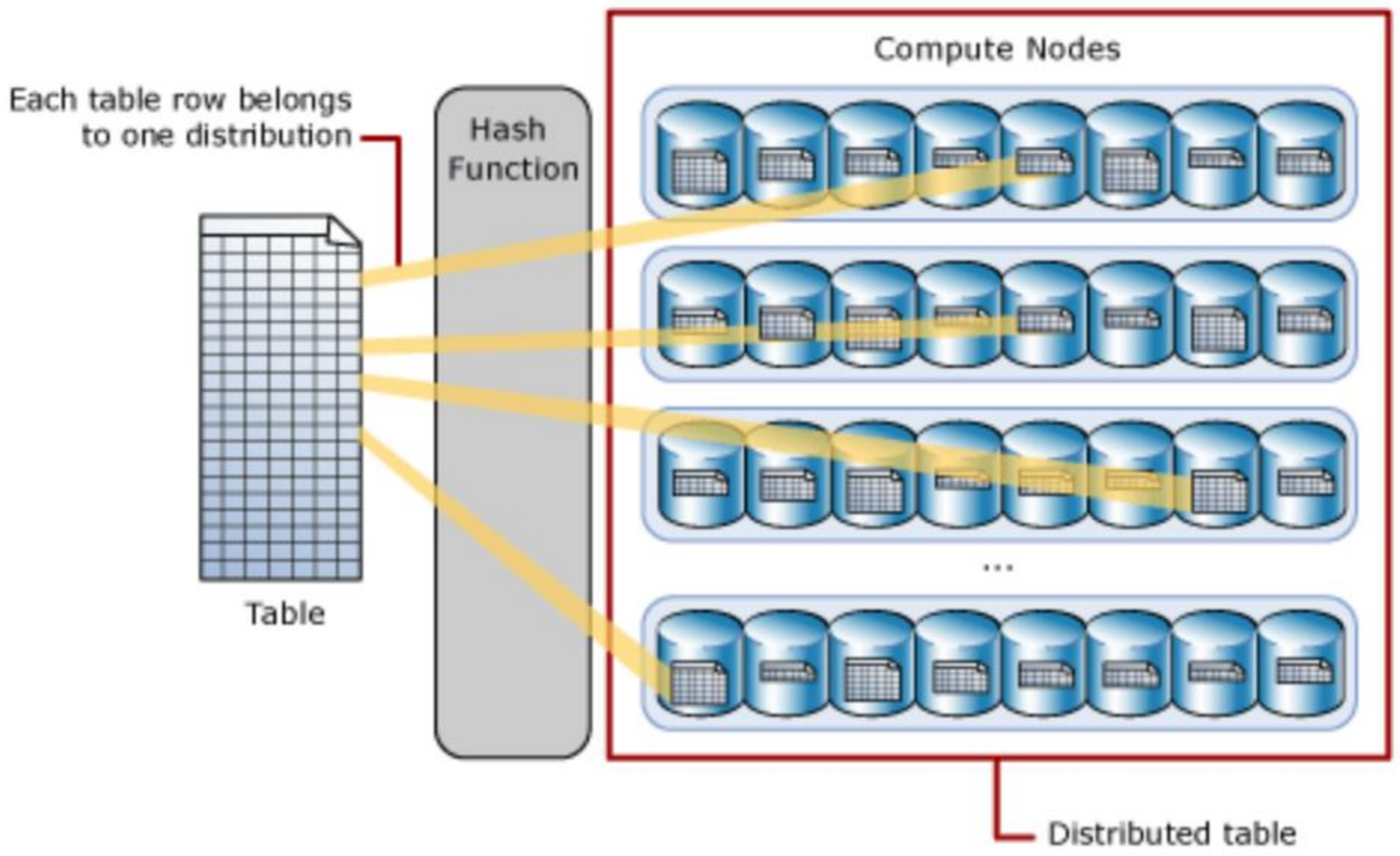
222 (Sum(Amt)) 111 (Sum(Amt)) 555 (Sum(Amt)) 333 (Sum(Amt))
444 (Sum(Amt))

HASH DISTRIBUTION

01



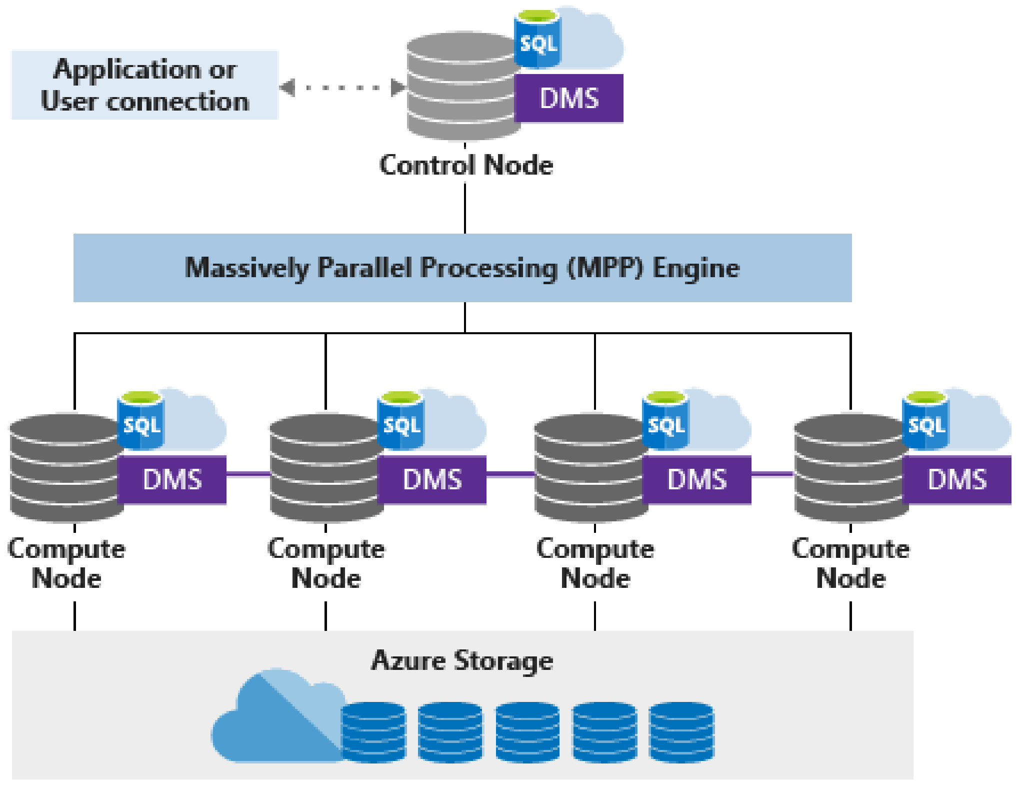
HASH DISTRIBUTION



https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/sql-data-warehouse-tables-distribute

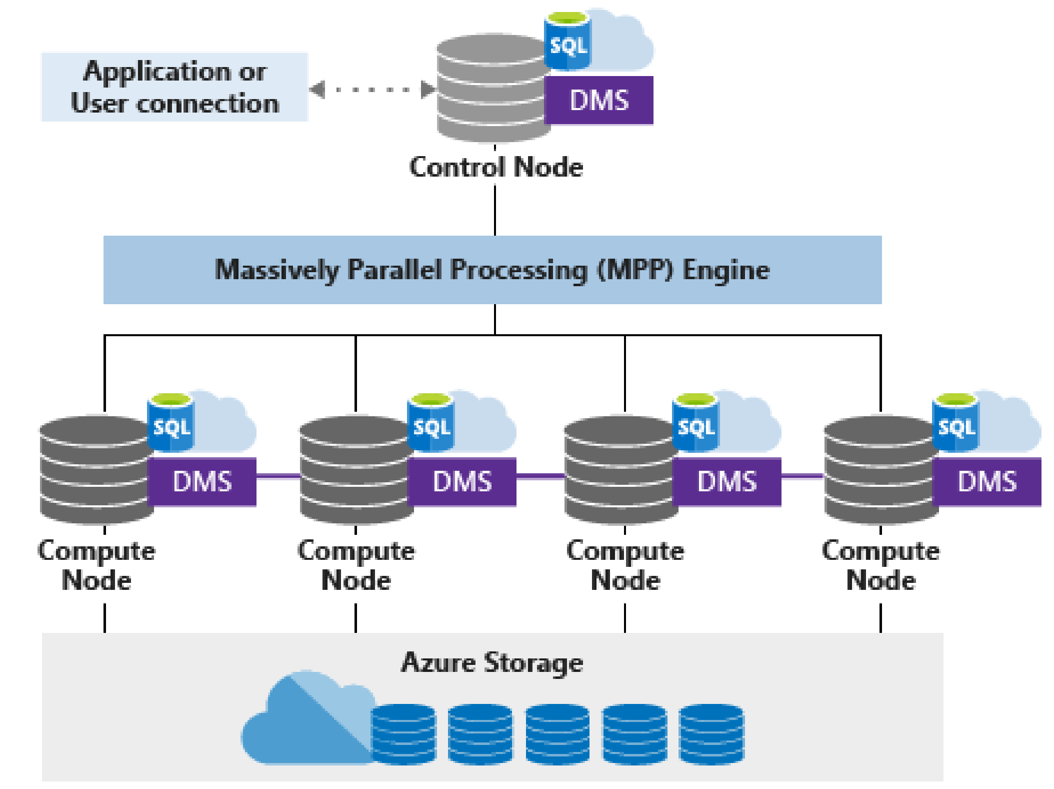
Synapse SQL – MPP Architecture

111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)



Synapse SQL – MPP Architecture

111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
111 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
444 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
222 (Amt, xxx, xxx)
555 (Amt, xxx, xxx)

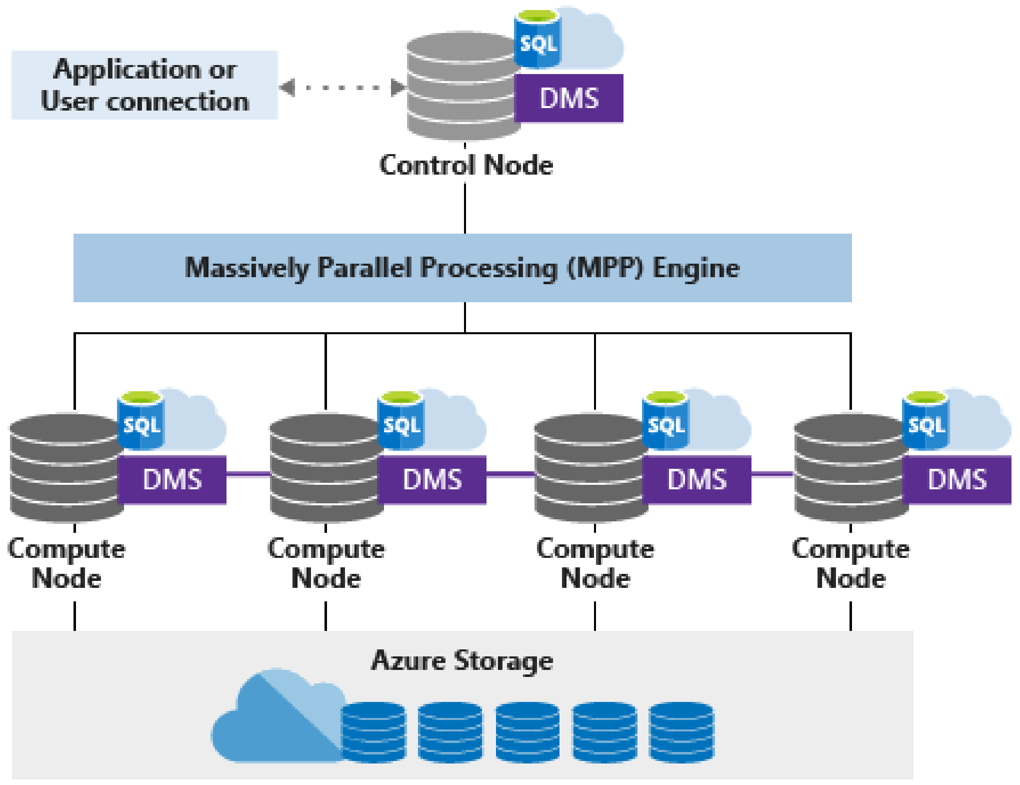


333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)
	222 (Amt, xxx, xxx)		444 (Amt, xxx, xxx)
			444 (Amt, xxx, xxx)
			444 (Amt, xxx, xxx)

Hash distribution based on ProductKey

Synapse SQL – MPP Architecture

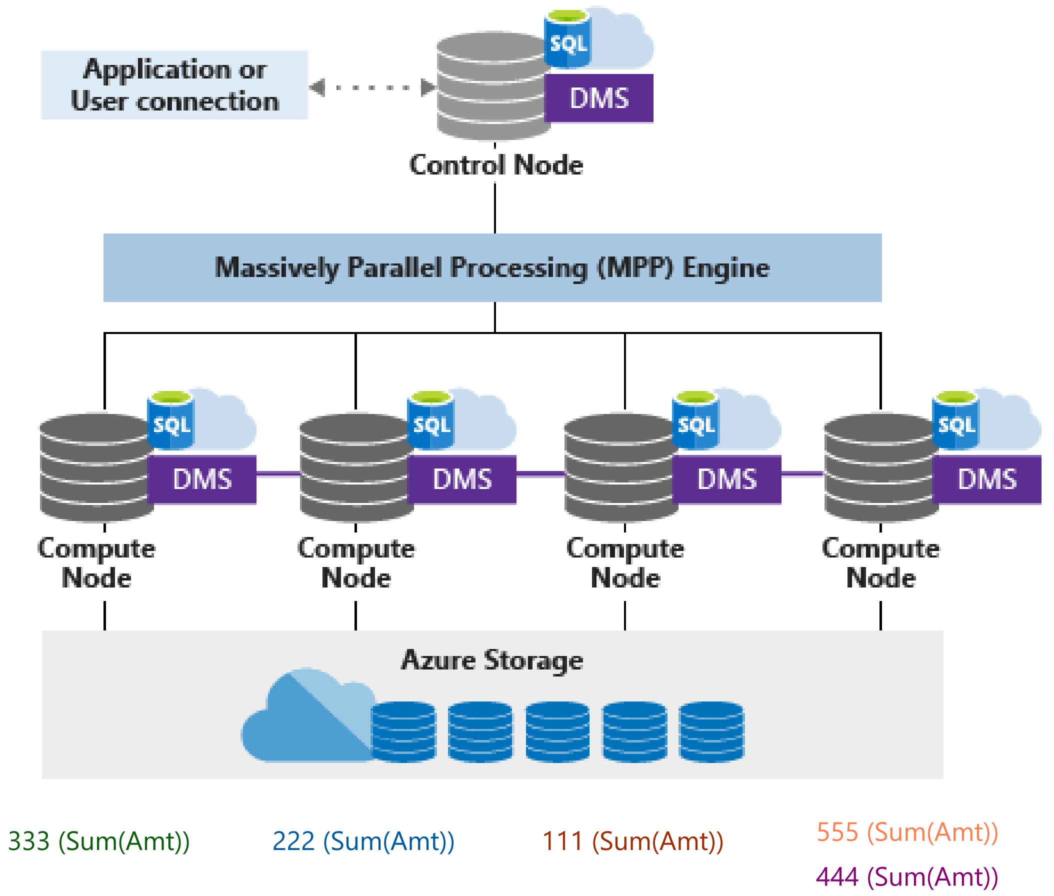
```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```



333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
333 (Amt, xxx, xxx)	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	555 (Amt, xxx, xxx)
	222 (Amt, xxx, xxx)	111 (Amt, xxx, xxx)	444 (Amt, xxx, xxx)
	222 (Amt, xxx, xxx)		444 (Amt, xxx, xxx)
			444 (Amt, xxx, xxx)
			444 (Amt, xxx, xxx)

Synapse SQL – MPP Architecture

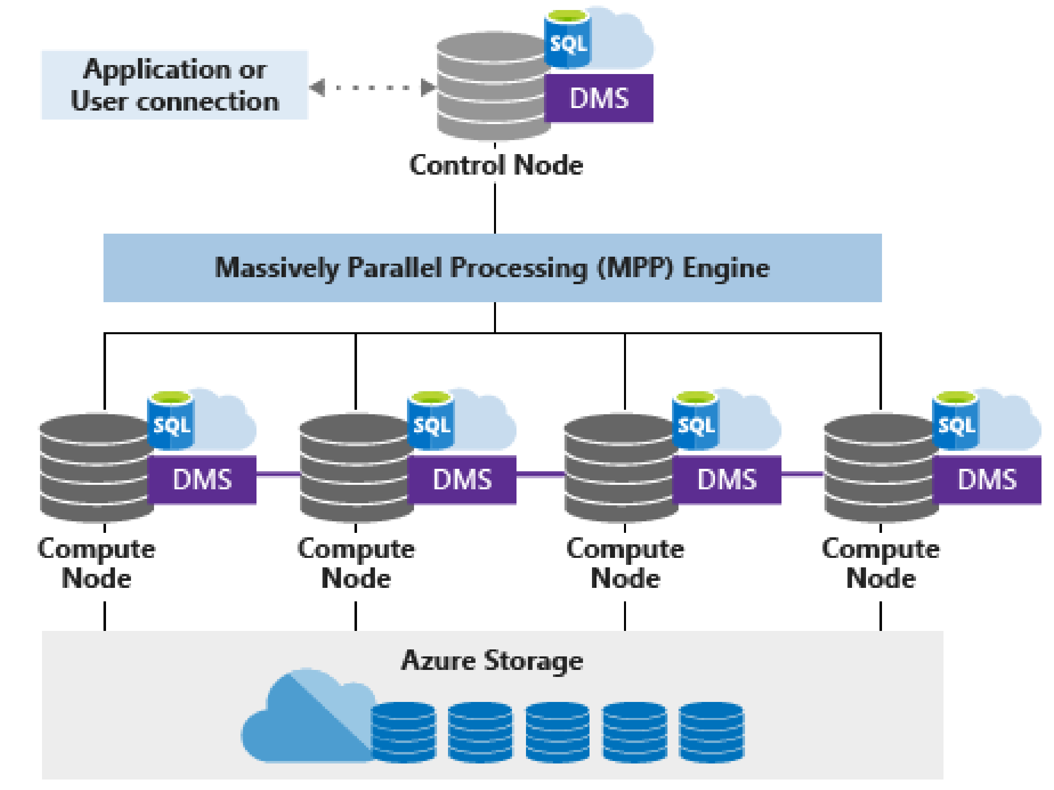
```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```



Synapse SQL – MPP Architecture

111 (Sum(Amt))
222 (Sum(Amt))
333 (Sum(Amt))
444 (Sum(Amt))
555 (Sum(Amt))

```
SELECT ProductKey, Sum(Amt)
FROM TableName
GROUP BY ProductKey
ORDER BY ProductKey
```



333 (Sum(Amt)) 222 (Sum(Amt)) 111 (Sum(Amt)) 555 (Sum(Amt))
444 (Sum(Amt))

Demo

Round-Robin distributed tables

- Distributes table rows evenly across all distributions
- Rows with equal values are not guaranteed to be assigned to the same distribution, resulting in more data movement
- Consider using the round-robin distribution for your table in the following scenarios:
  - When getting started as a simple starting point since it is the default
  - If there is no obvious joining key
  - If there is not good candidate column for hash distributing the table
  - If the table does not share a common join key with other tables
  - If the join is less significant than other joins in the query
  - When the table is a temporary staging table

Hash distributed table

- Hash function is used to assign each row to one distribution – deterministic hash function
- Identical values always hash to the same distribution
- Distribution column should have an even distribution of values
- Hash-distributed tables work well for large fact tables
- Consider using a hash-distributed table when:
  - The table size on disk is more than 2 GB.
  - The table has frequent insert, update, and delete operations.

Selecting a right distribution key

For large fact tables, best option is to Hash Distribute

- All of the distributions should have approximately the same number of rows
- Distribute on column that is joined to other fact tables
- Primary or surrogate key

However, be mindful of …

- Hash column should have highly distinct values (Minimum >60 distinct values)
- Avoid distributing on a date column
- Avoid distributing on column with high frequency of NULLs and default values (e.g. -1)
- Distribution column is NOT updatable. Updates to data in the distribution column could result in data shuffle operation.
- For compatible joins use the same data types for two distributed tables

Selecting a right distribution key

Choose a distribution key that minimizes data movement

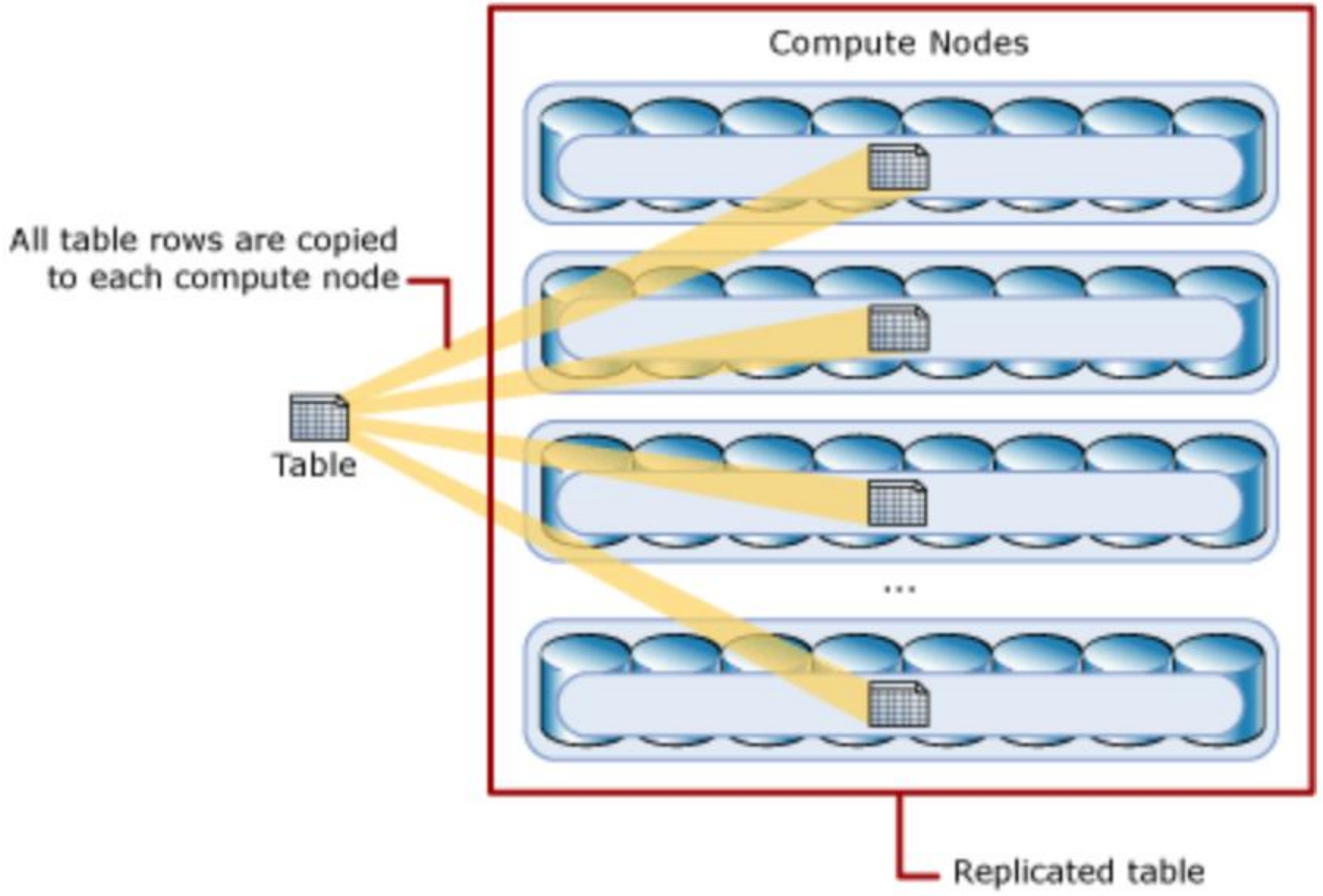
- Data movement commonly happens when queries have joins and aggregations
- Distribution key - JOIN, GROUP BY, DISTINCT, OVER, and HAVING clauses
- Is not used in WHERE clauses. This could narrow the query to not run on all the distributions.
- Is not a date column. WHERE clauses often filter by date. When this happens, all the processing could run on only a few distributions.

If there are no distribution columns that make sense, then use Round Robin as last resort

Replicated Tables

- Replicated table has a full copy of the table on each Compute node
- Replicating a table removes the need to transfer data among Compute nodes before a join or aggregation.
- Ideal for small dimension tables (<2 GB compressed) – not a hard limit though, if the data is static and does not change, you can replicate larger tables.
- Replicated tables may not yield the best query performance when:
  - The table has frequent insert, update, and delete operations. These DML operations require a rebuild of the replicated table. Rebuilding frequently can cause slower performance.
  - The data warehouse is scaled frequently. Scaling a data warehouse changes the number of Compute nodes, which incurs a rebuild.
  - The table has a large number of columns, but data operations typically access only a small number of columns

Replicated Tables



https://docs.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/design-guidance-for-replicated-tables

Dimension Table

Small dimension table (< 60M rows)

- Clustered index
- Round Robin
- Replicated tables

Large dimension table

- Same design as fact table
- Clustered columnstore (by default) and distribute on join key

Demo