

DP-203 Microsoft Azure Data Engineer

Day 10 – Azure Synapse Analytics (cont...)

3rd Aug 2021

Vinodkumar Bhovi

Introduction

MPP or Massive Parallel Processing

Storage & Data Distribution (Hash, Round-robin, Replicate)

Data types and Table types (Columnstore, Heap, Clustered B-tree index)

Partitioning and Distribution key

Applications in Dimensional modeling

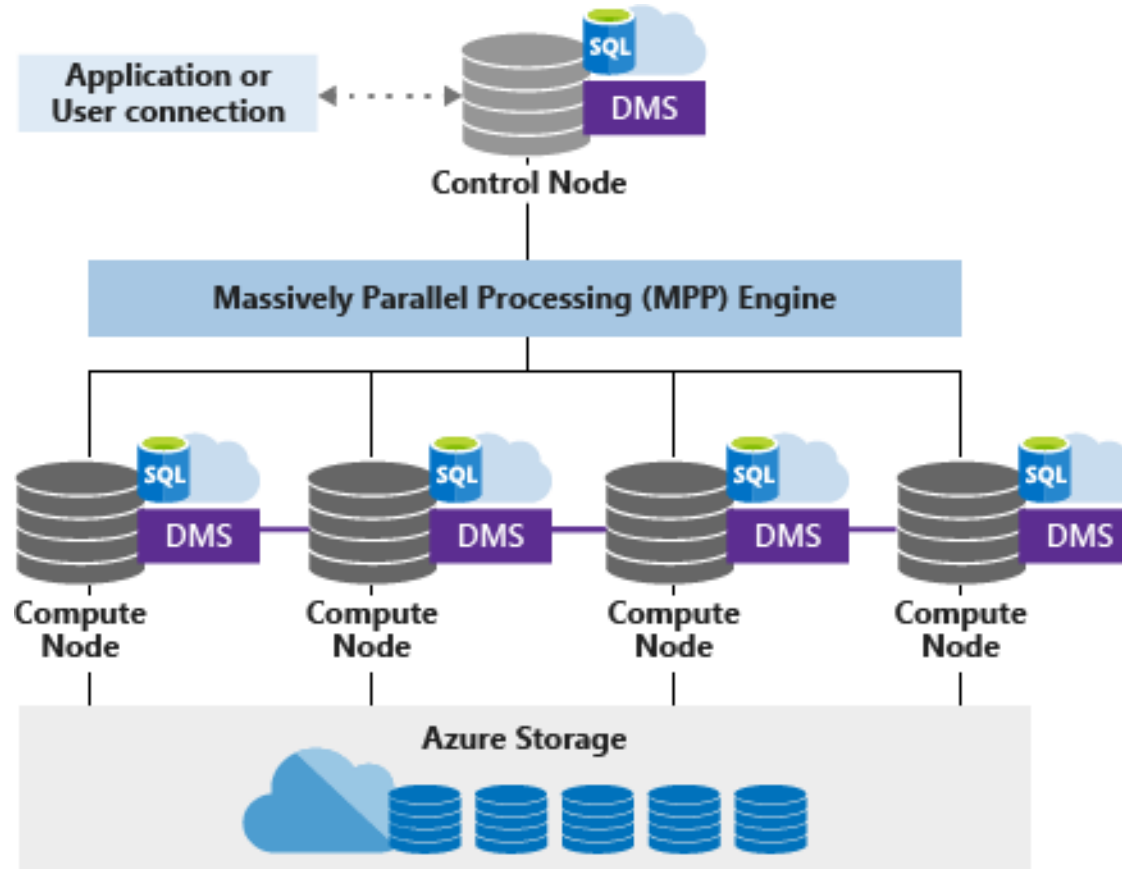
Demo – Table Analysis before Migration to Cloud



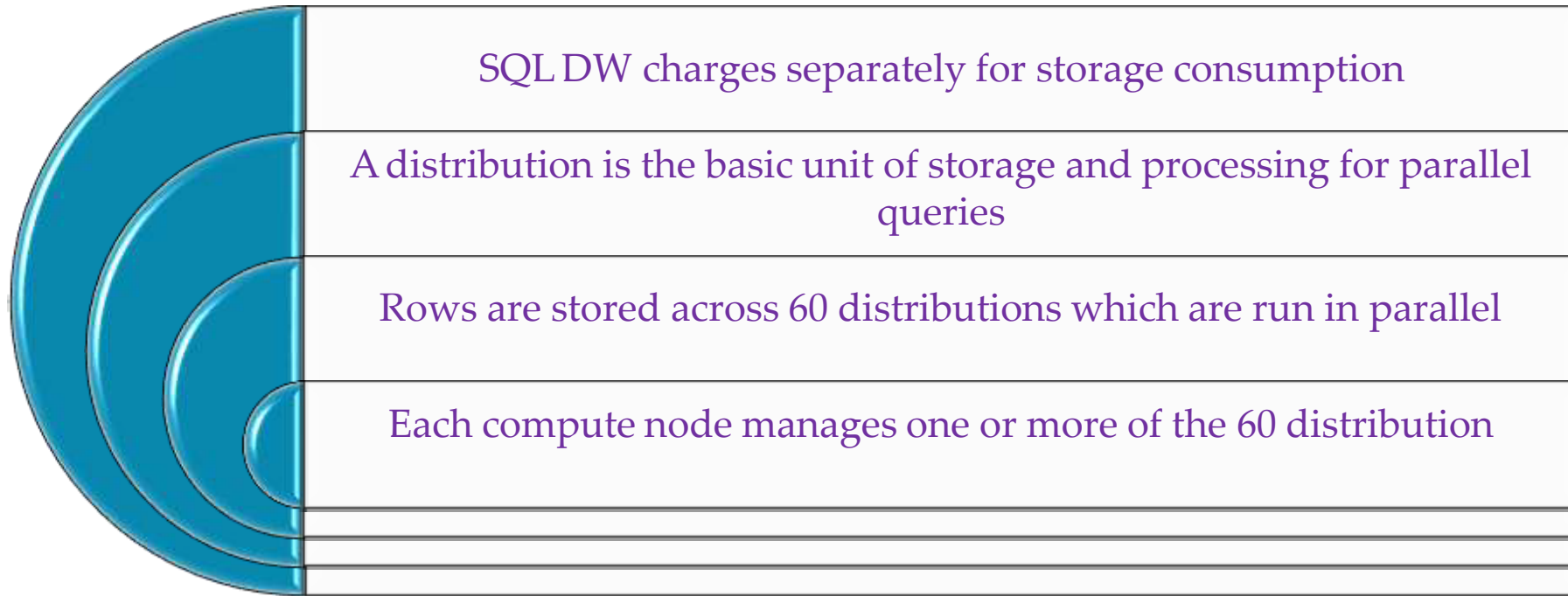
Azure
Synapse
Analytics

Azure Synapse MPP Architecture

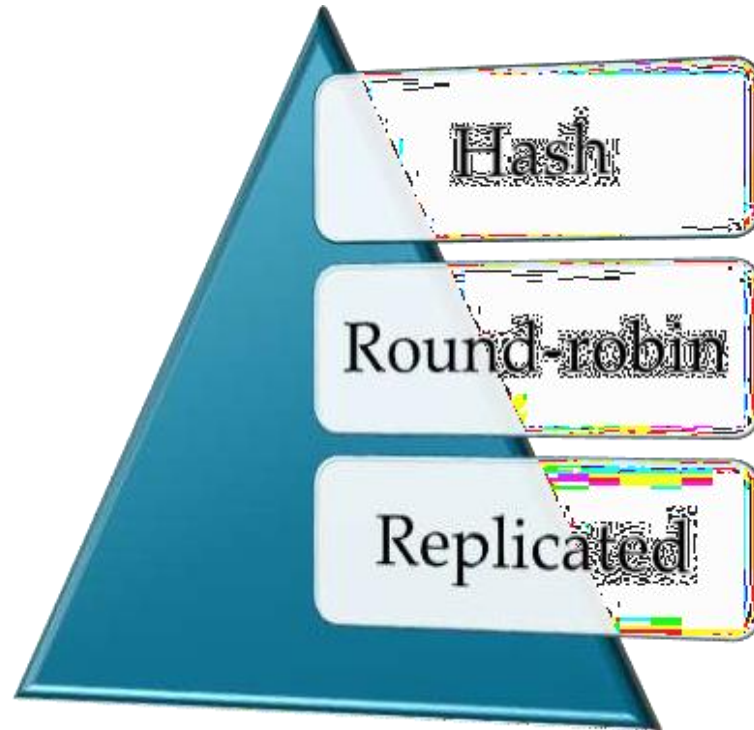
DWU	Loading 3 Tables	Ran Report
100	15	20
500	3	4



Azure Storage and Distribution



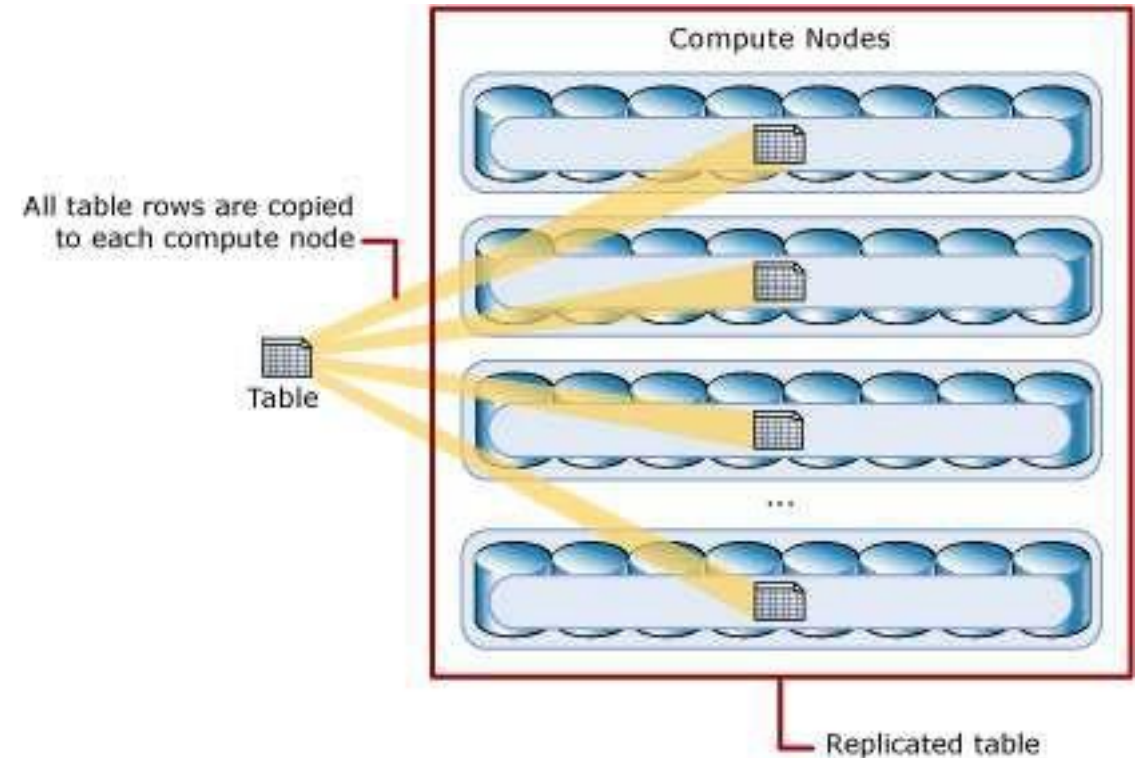
Sharding Patterns



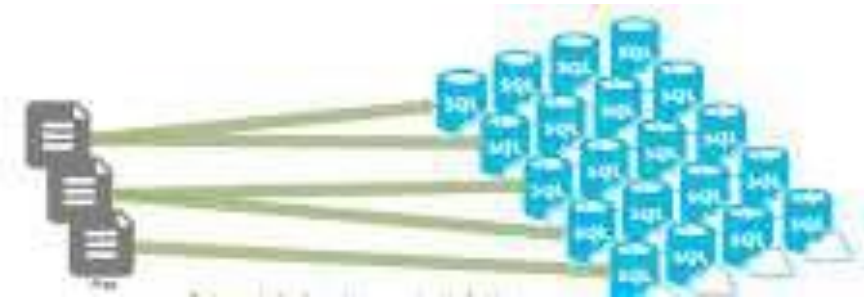
Replicated Tables

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

```
CREATE TABLE [dbo].[BusinessHierarchies](  
    [BookId] [nvarchar](250) ,  
    [Division] [nvarchar](100) ,  
    [Cluster] [nvarchar](100) ,  
    [Desk] [nvarchar](100) ,  
    [Book] [nvarchar](100) ,  
    [Volcker] [nvarchar](100) ,  
    [Region] [nvarchar](100)  
)  
WITH  
(  
    CLUSTERED COLUMNSTORE INDEX  
    DISTRIBUTION = REPLICATE  
)  
;
```



Round Robin tables

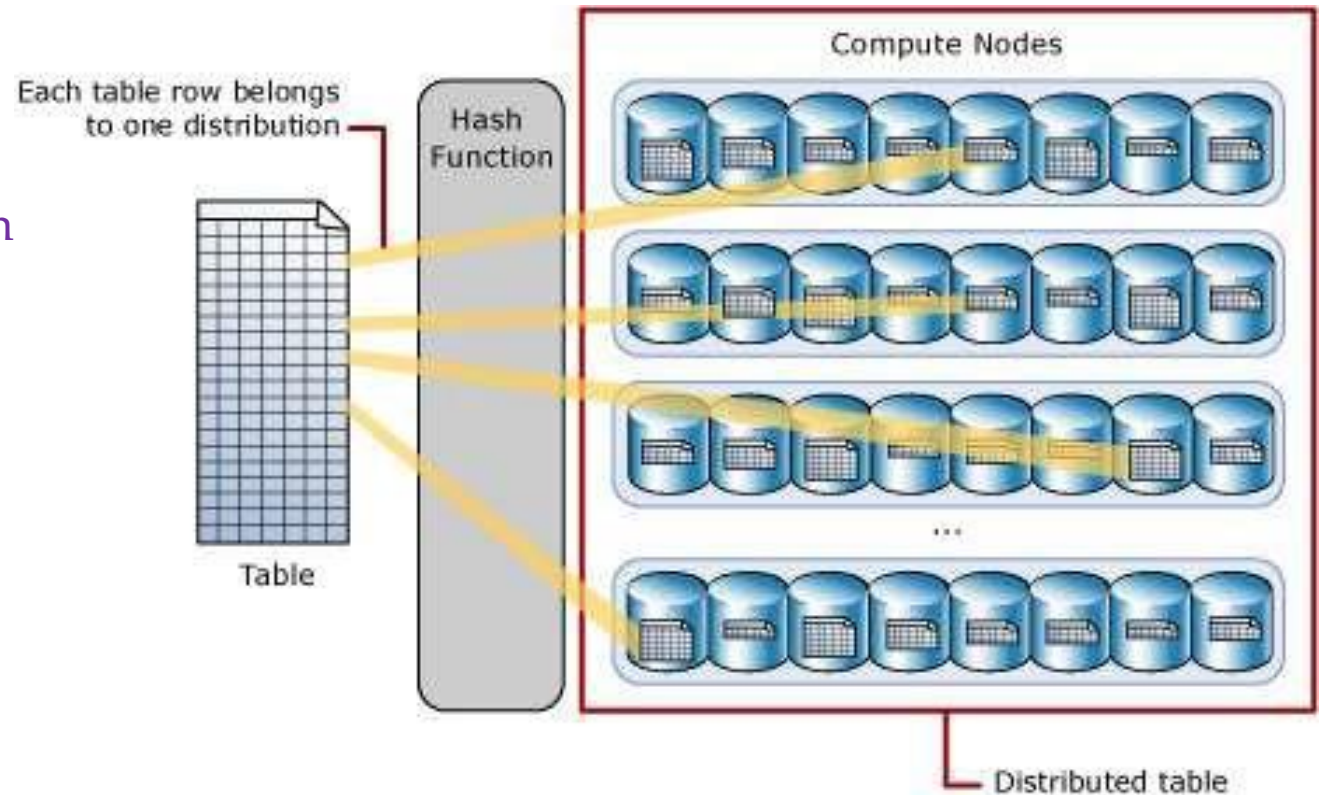


- Generally use to load staging tables
- Distribute data evenly across the table without additional optimization
- Joins are slow, because it requires to reshuffle data
- Default distribution type

```
CREATE TABLE [dbo].[Dates](  
    [Date] [datetime2](3) ,  
    [DateKey] [decimal](38, 0) ,  
    ..  
    ..  
    [WeekDay] [nvarchar](100) ,  
    [Day Of Month] [decimal](38, 0)  
)  
  
WITH  
{  
    CLUSTERED COLUMNSTORE INDEX  
    DISTRIBUTION = ROUND_ROBIN  
}  
;
```

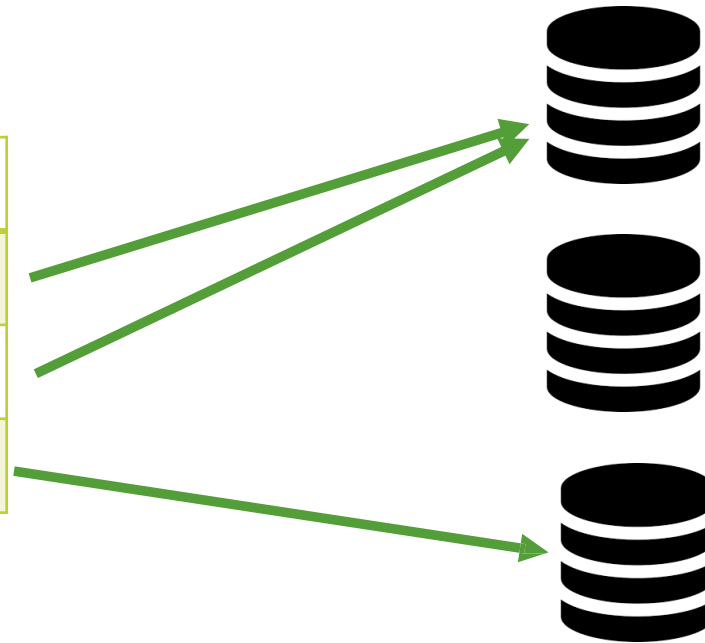
Hash Distribution Tables

- Highest performance for large tables
- Each row belong to one particular distribution
- It is used mostly for larger tables



Hash Distribution Tables

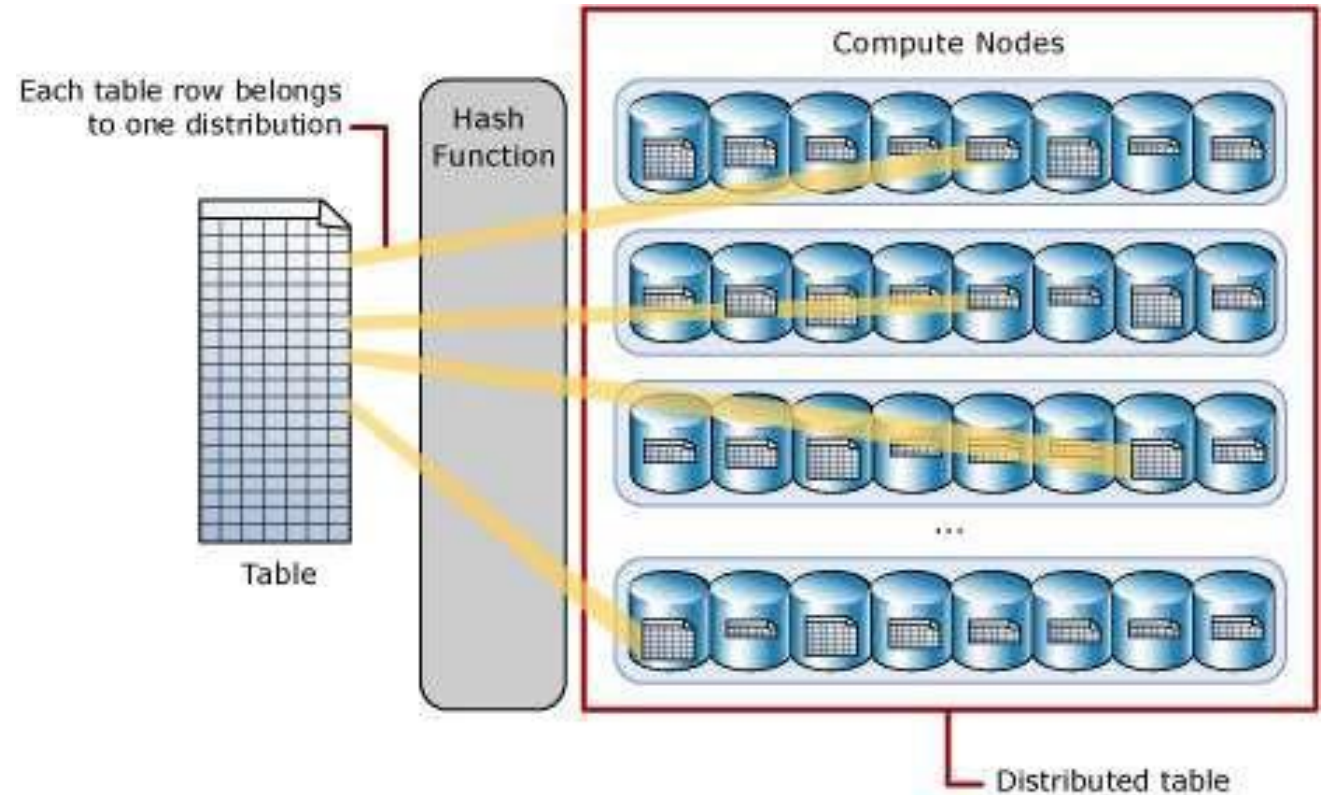
Record	Product	Store
1	Soccer	New York
2	Soccer	Los Angeles
3	Football	Phoenix



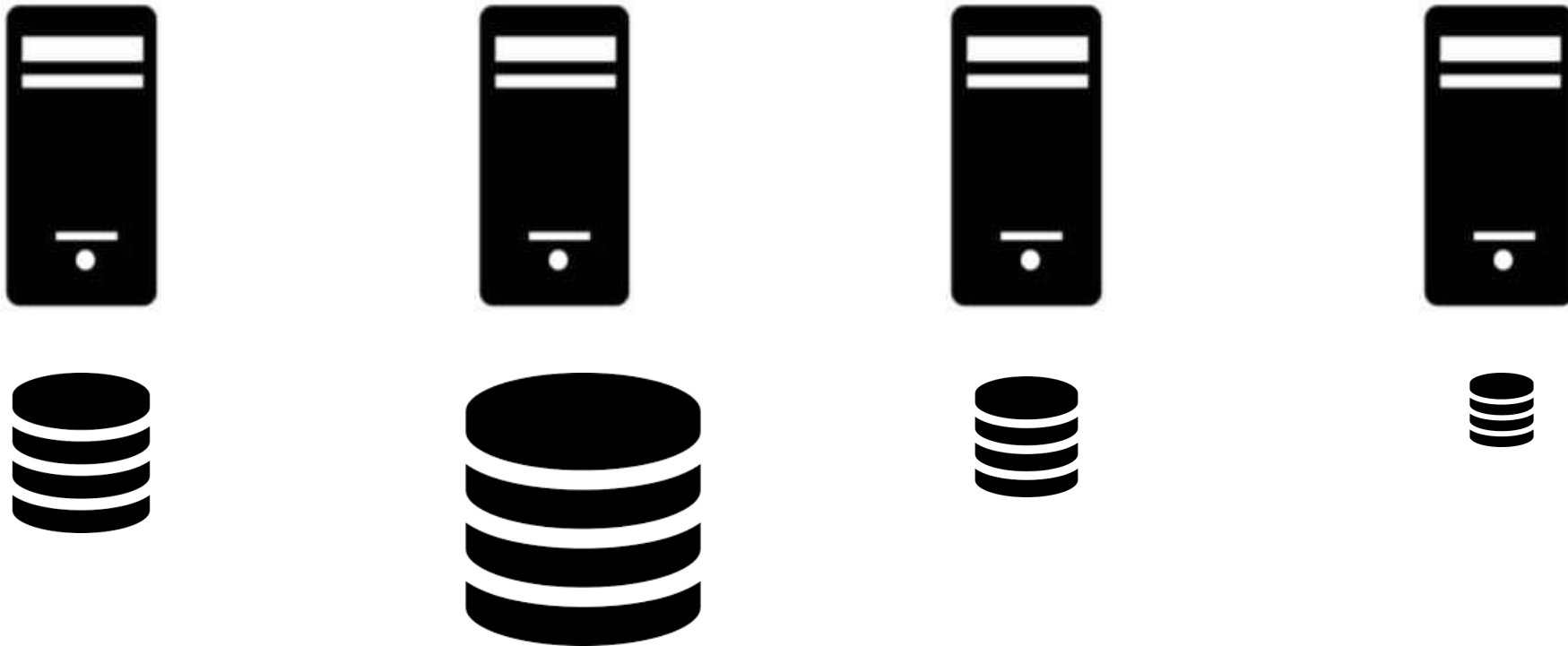
Hash Distribution Tables

- Highest performance for large tables
- Each row belong to one particular distribution
- It is used mostly for larger tables

```
CREATE TABLE [dbo].[EquityTimeSeriesData](  
    [Date] [varchar](30),  
    [BookId] [decimal](38, 0),  
    [P&L] [decimal](31, 7),  
    [VaRLower] [decimal](31, 7)  
)  
WITH  
(  
    CLUSTERED COLUMNSTORE INDEX  
    , DISTRIBUTION = HASH([P&L])  
)  
;
```



Avoid Data Skew



Even Distribution

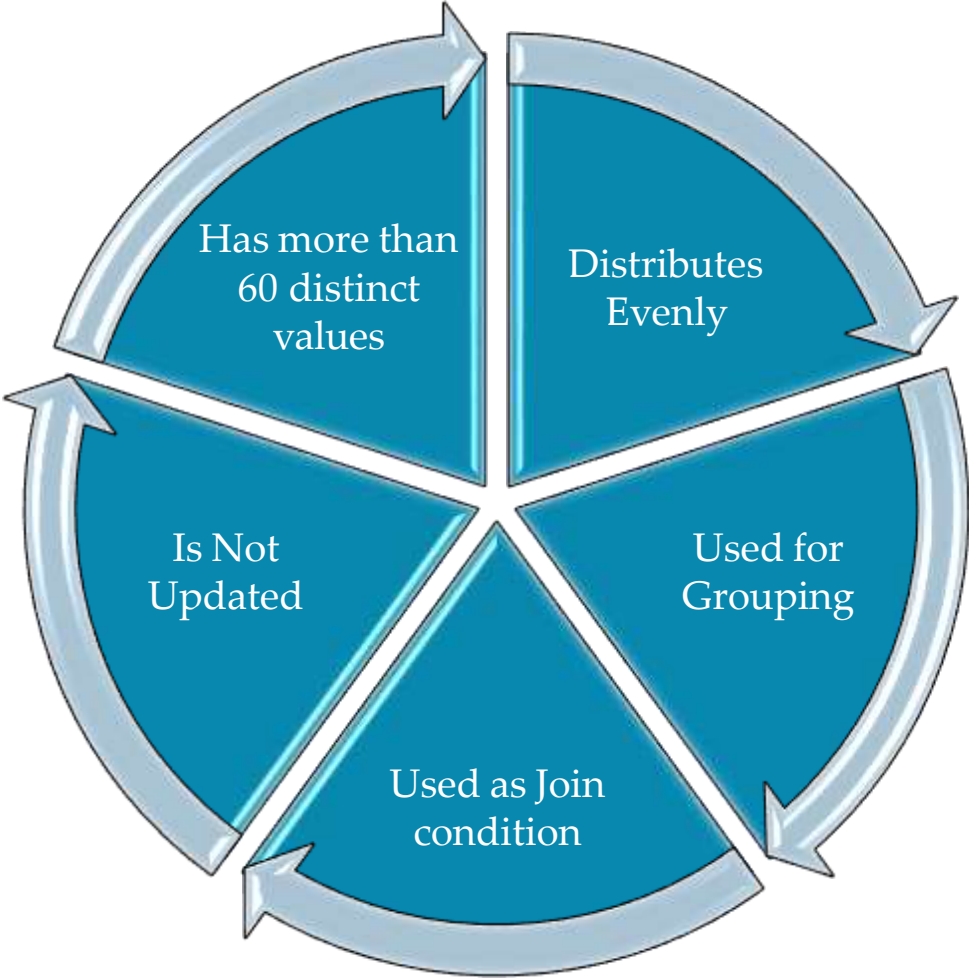


Distribution key

Determines the method in which Azure SQL Data Warehouse spreads the data across multiple nodes.

Azure SQL Data Warehouse uses up to 60 distributions when loading data into the system.

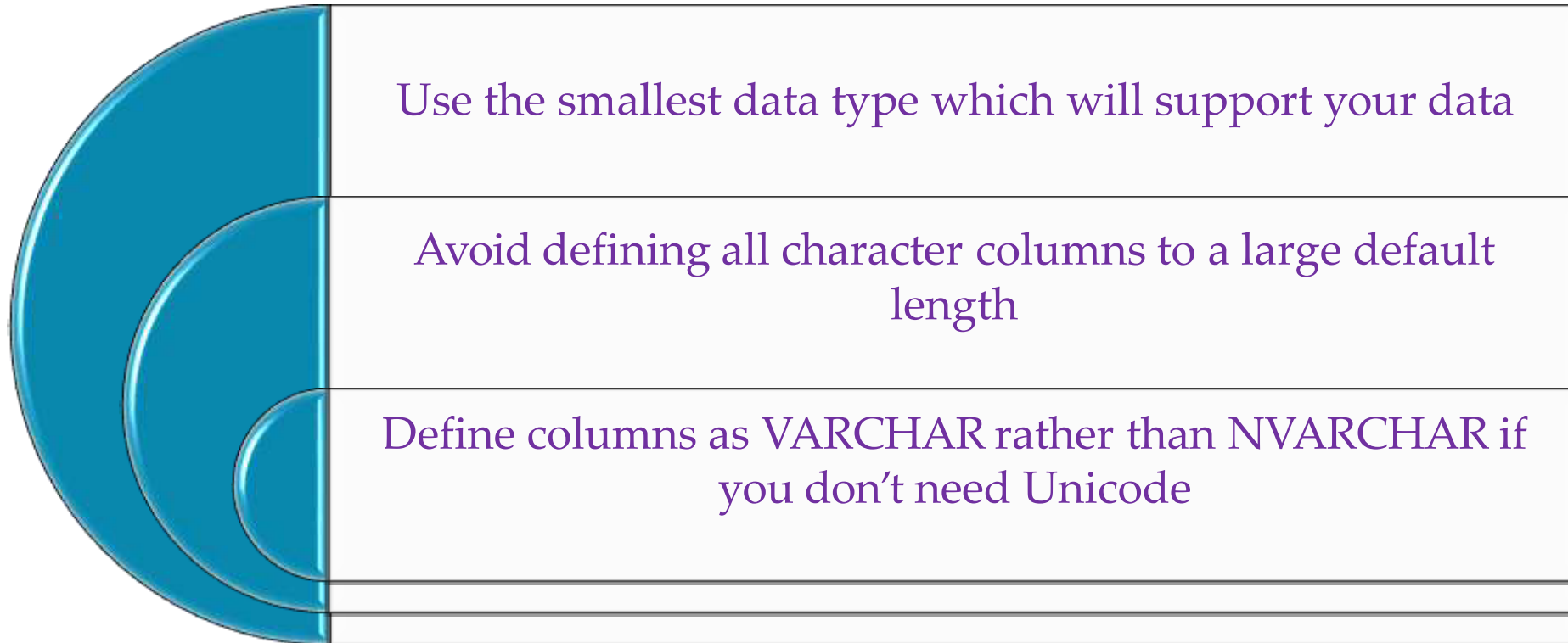
Good Hash Key



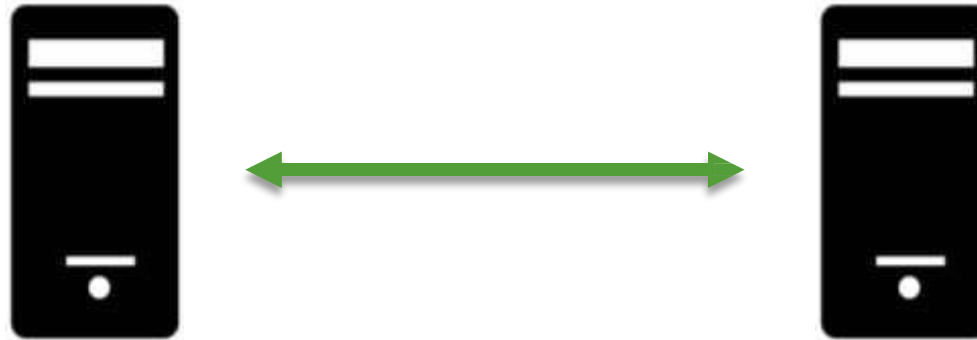
What Data Distribution to Use?

Type	Great fit for	Watch out if...
Replicated	Small-dimension tables in a star schema with less than 2GB of storage after compression	<ul style="list-style-type: none">• Many write transaction 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)	<ul style="list-style-type: none">• Temporary/Staging table• No obvious joining key or good candidate column.	Performance is slow due to data movement
hash	<ul style="list-style-type: none">• Fact tables• Large dimension tables	The distribution key can't be updated

Data types



Data types



The goal is to not only save space but also move data as efficiently as possible.

Data types



Some complex data types (XML, geography, etc) are not supported on Azure SQL Data Warehouse yet.

Table types

Clustered columnstore

- Updateable primary storage method
- Great for read-only

Default for synapse and recommended

Heap

- Data is not in any particular order.
- Use when data has no natural order.

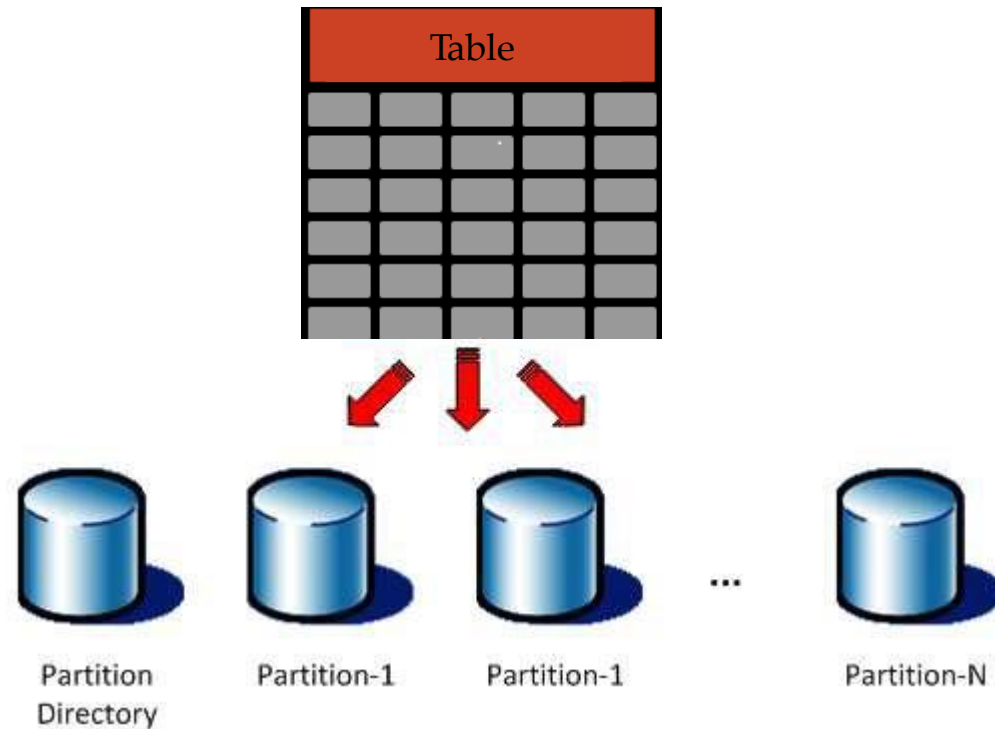
Loads faster and no index

Clustered Index

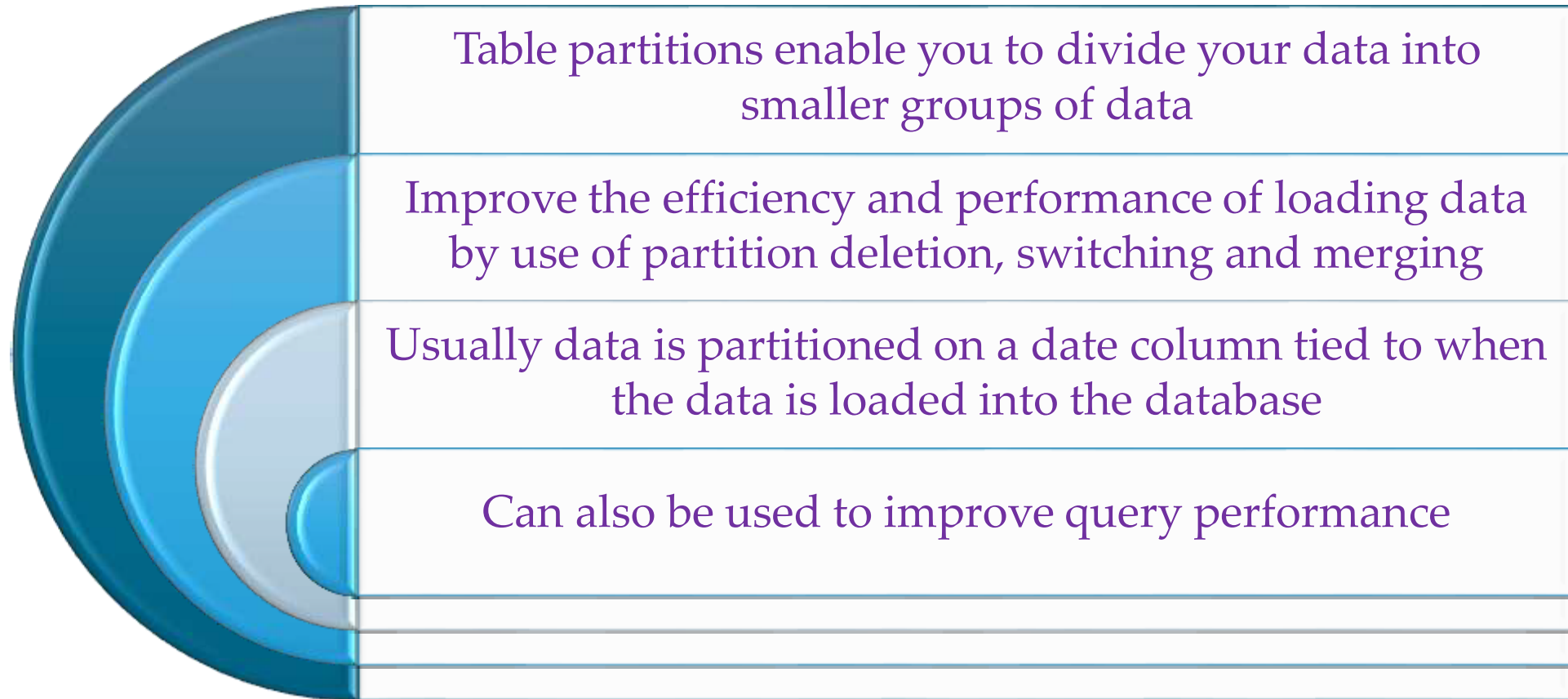
- An index that is physically stored in the same order as the data being indexed

On prem index method, you can specify the column name

Table Partitioning



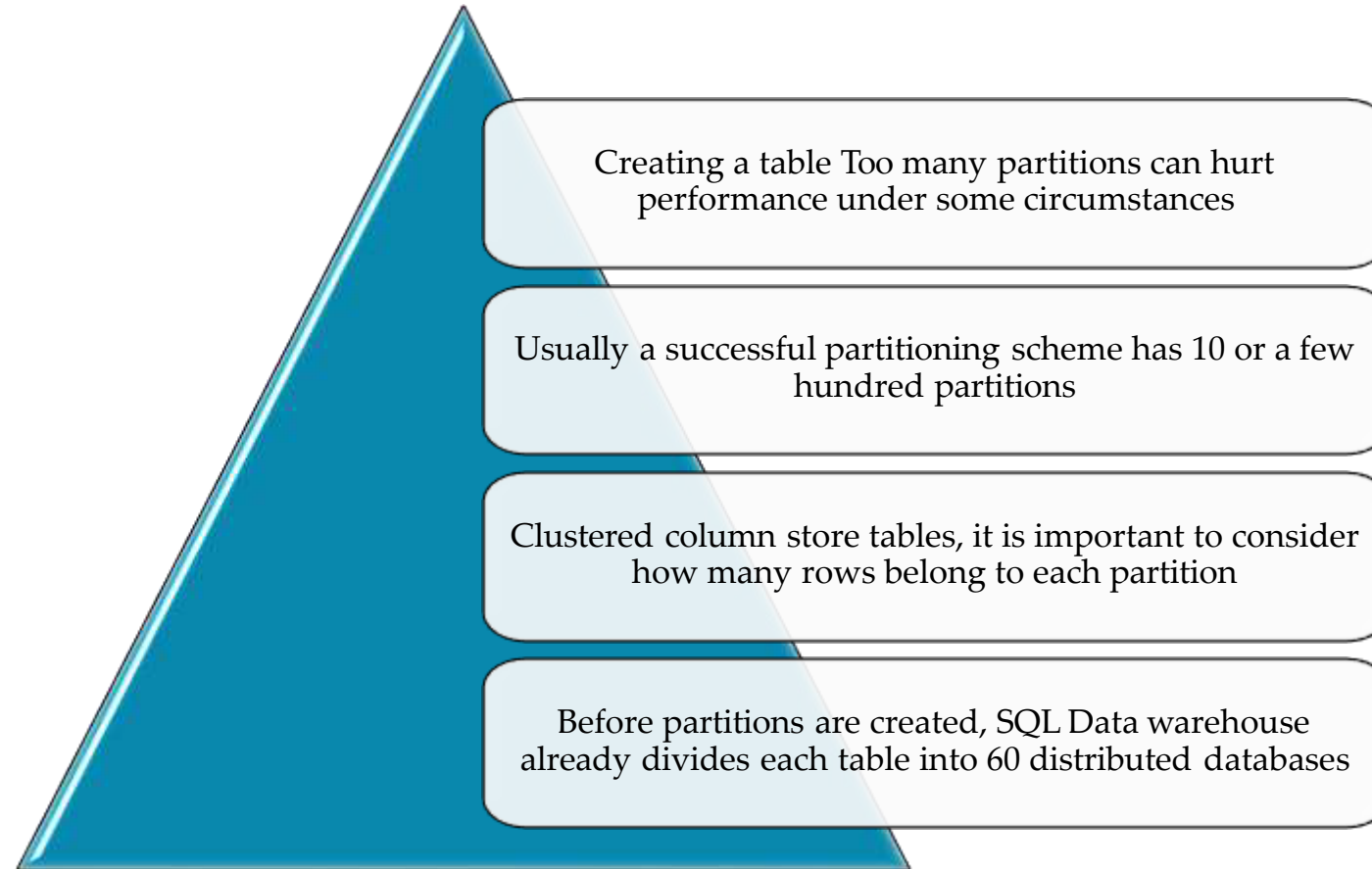
Partitioning



Why Partitioning?



Partitions best practices





A highly granular partitioning scheme can work in SQL Server but hurt performance in Azure SQL Data Warehouse.

Example

60 Distributions  365 Partitions  21900 Data Buckets

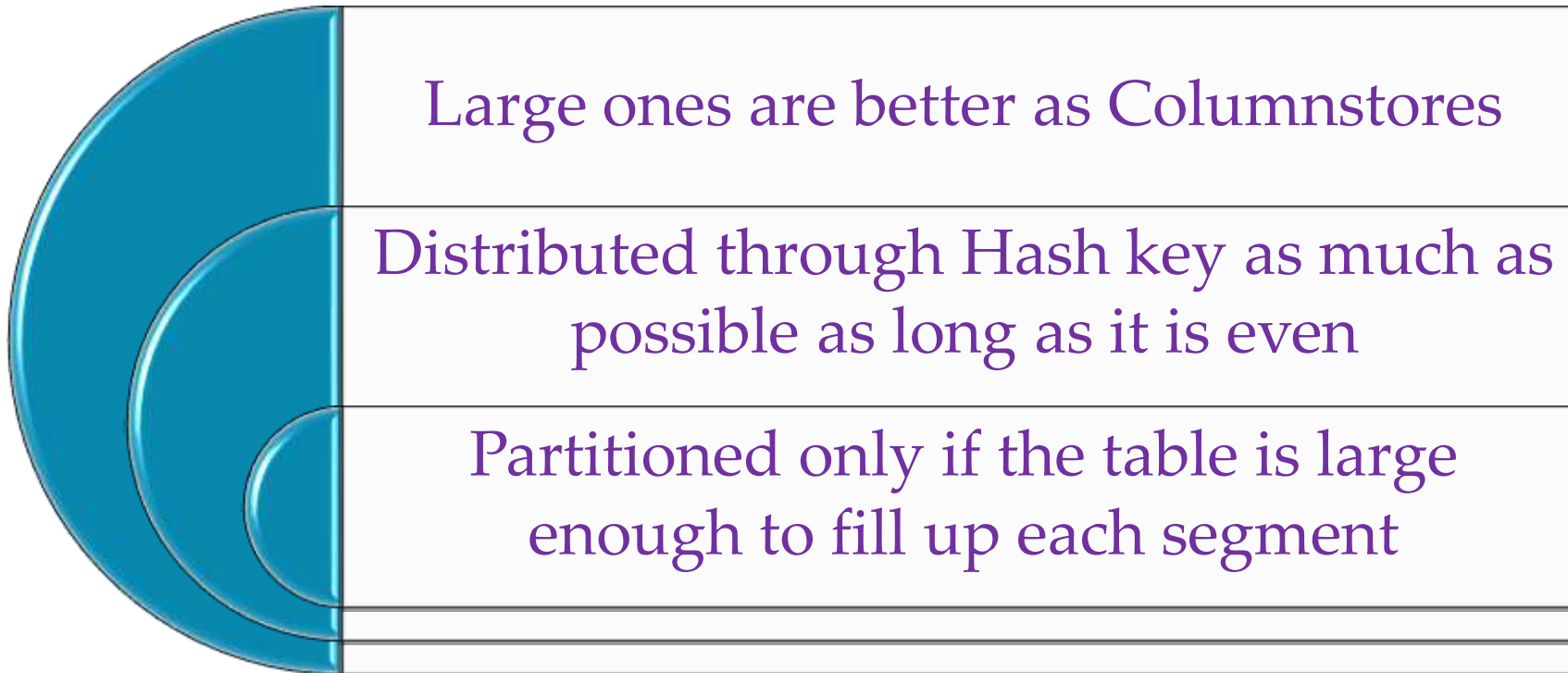
21900 Data Buckets  Ideal Segment Size (1M Rows)  21 900 000 000 Rows




Lower Granularity (week, month)
can perform better depending on
how much data you have.

How do we apply these principles to a
Dimensional model?

Fact Tables



Dimension Tables



Can be Hash distributed or Round-Robin if there is no clear candidate join key
Columnstore for large dimensions
Heap or Clustered Index for small dimensions
Add secondary indexes for alternate join columns
Partitioning not recommended

Workload Management

- Workload management ensures that enough resources exist for data warehousing activities like loading, transforming and querying data

Workload Management

- Workload Classification
- Workload Importance
- Workload Isolation

Workload Classification

- Assign users to a role that has a corresponding resource class
- CREATE WORKLOAD CLASSIFIER
- A classification that is based on a database user will take the precedence over a role membership

Workload Importance

- Workload importance influences the order in which a request gets access to resources
- 5 Level of importance – Low, below_normal, normal, Above_normal, & high
- A request with a higher importance will run before a request with lower importance

Workload Isolation

- Workload isolation reserves resources for a workload group and use workload groups to define isolation
- Define min & max level of concurrency

Summary

MPP or Massive Parallel Processing

Billing = Compute + Storage

Data Distribution (Hash, Round-robin, Replicate)

Data types and Table types

Partitioning Data

Best practice – Fact and Dimension table design

Demo – Analyse Data Distribution



Azure
Synapse
Analytics