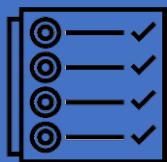




## Implement Relational Data Stores

# Implement relational data stores



## Learning Objectives

- Azure SQL Database offerings
  - PaaS vs IaaS
  - PaaS deployment options
  - Purchasing models and Service Tier
  - Elastic Database
  - Managed Database
  - Security
- Azure SQL Datawarehouse
  - Traditional vs Modern Architecture
  - Azure Synapse Analytics
  - MPP Architecture
  - Storage and Sharding Pattern
  - Data Distribution and Table types
  - Partitioning
- Loading methods
  - PolyBase vs SSIS
  - PolyBase steps and demos
- Azure SQL Server vs Datawarehouse

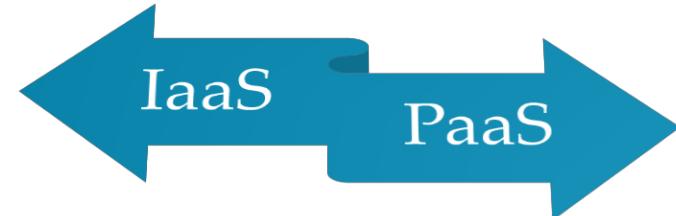
# Azure SQL Database

Eshant Garg

Advisor, Data Specialist

[Eshant.garg@gmail.com](mailto:Eshant.garg@gmail.com)

# Introduction



## Azure SQL Database Deployment Types



**Single Database**  
Own set of resources



**Elastic Pool**  
Collection of databases sharing resources



**Managed Instance**  
Dedicated engine instance running collection of databases

# Why SQL Server in Azure?



Fully Managed



Predictable  
performance  
and pricing



Elastic pool for  
unpredictable  
workloads



99.99%  
availability  
built-in



Geo-replication  
and restore  
services



Supports existing SQL  
Server tools, libraries,  
and APIs



Scalability with no  
downtime



Secure and compliant  
for your sensitive data

# Azure IaaS vs PaaS Database offerings?



**SQL Server on Azure VMs**  
SQL Server inside a  
fully-managed VM in Azure



**Azure SQL Database**  
Database-as-a-service (DBaaS)  
hosted in Azure

# Responsibility comparison



**SQL Server on Azure VMs**  
SQL Server inside a  
fully-managed VM in Azure



**Azure SQL Database**  
Database-as-a-service (DBaaS)  
hosted in Azure

# Benefits comparison



**SQL Server on Azure VMs**  
SQL Server inside a  
fully-managed VM in Azure



**Azure SQL Database**  
Database-as-a-service (DBaaS)  
hosted in Azure

# Limitations comparison

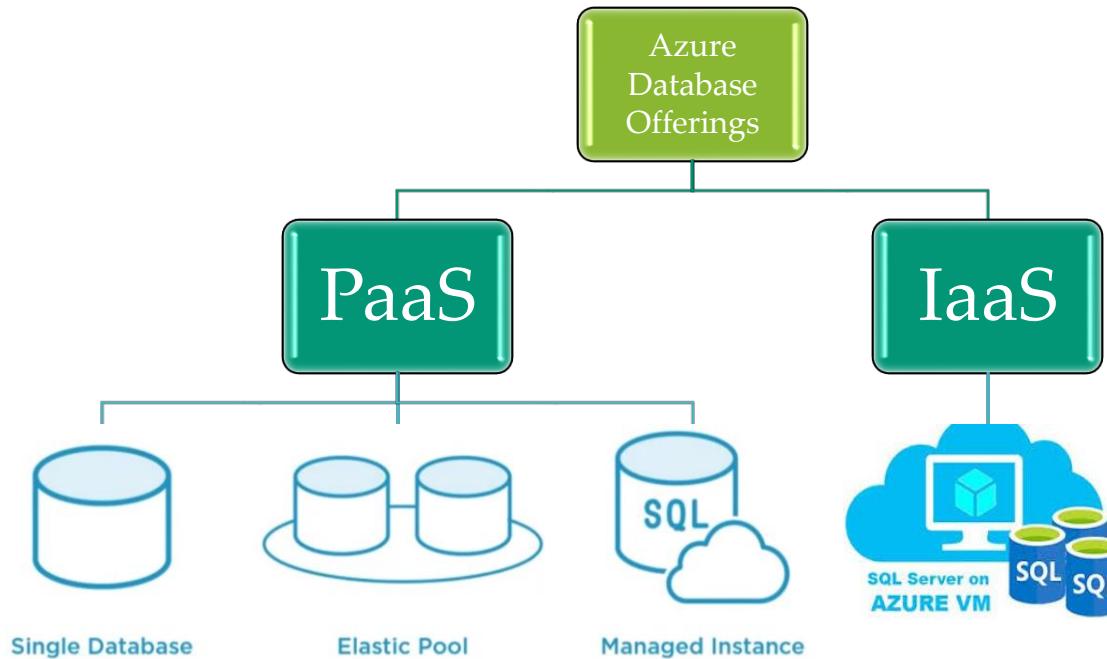


**SQL Server on Azure VMs**  
SQL Server inside a  
fully-managed VM in Azure



**Azure SQL Database**  
Database-as-a-service (DBaaS)  
hosted in Azure

# Azure Database Deployment options



# SQL Server(PaaS) Deployment Options



## Single database

Each DB with its own guaranteed compute, memory, and storage



## Elastic pool

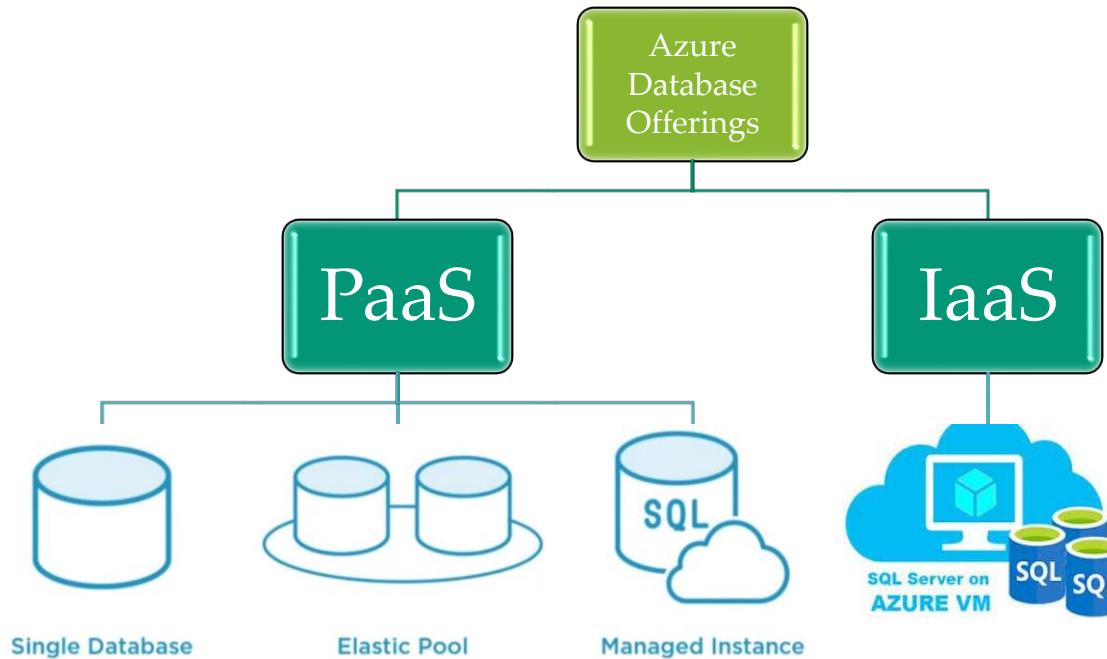
Fixed resources will be shared by all databases in the pool



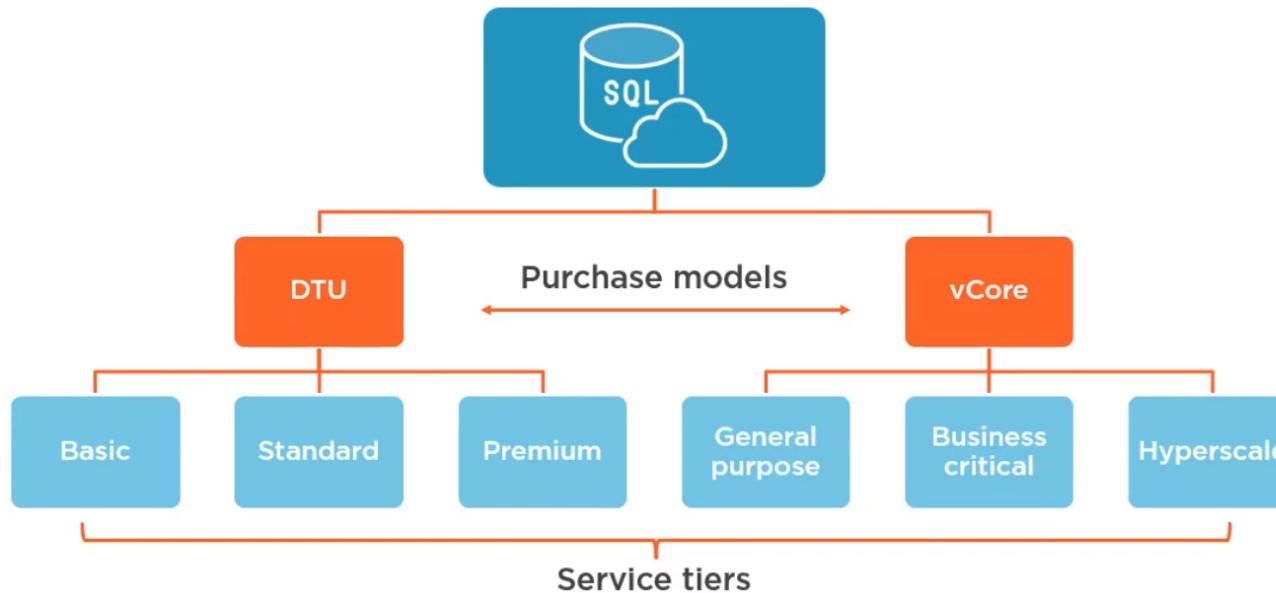
## Managed instance

Each managed instance has its guaranteed resources

# Azure Database Deployment options



# Azure SQL Database Service Tiers



# vCore-based vs DTU-based Model

## vCore-based

- For Single database, elastic pool and managed instance
- Best for customers who need flexibility, control, and transparency
- Straightforward way to translate on-premises workload to the cloud
- Microsoft recommends vCore-based model

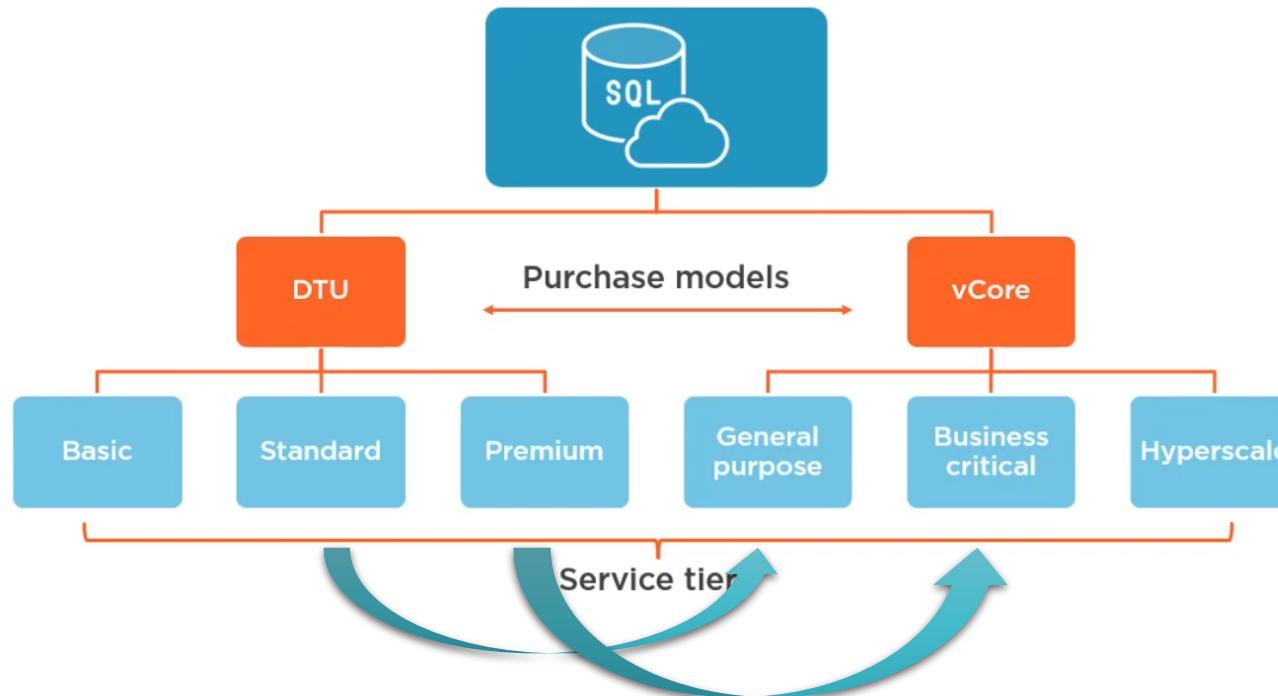
## DTU-based

- Only for single database and elastic pool
- Best for customers who want single, preconfigured resource options
- Might need to calculate the needed DTUs before migration
- If the DTU-based purchasing model needs your performance and business requirements, you should continue using it.

# Converting DTU-based Model to vCore-based

- If your single database or elastic pool consumes more than 300 DTUs, converting to the vCore-based model might reduce your costs
- You can use API of your choice or Azure Portal to convert to vCore based model with 'no downtime'.
- Azure SQL Database managed instance only supports vCore-based purchasing model.

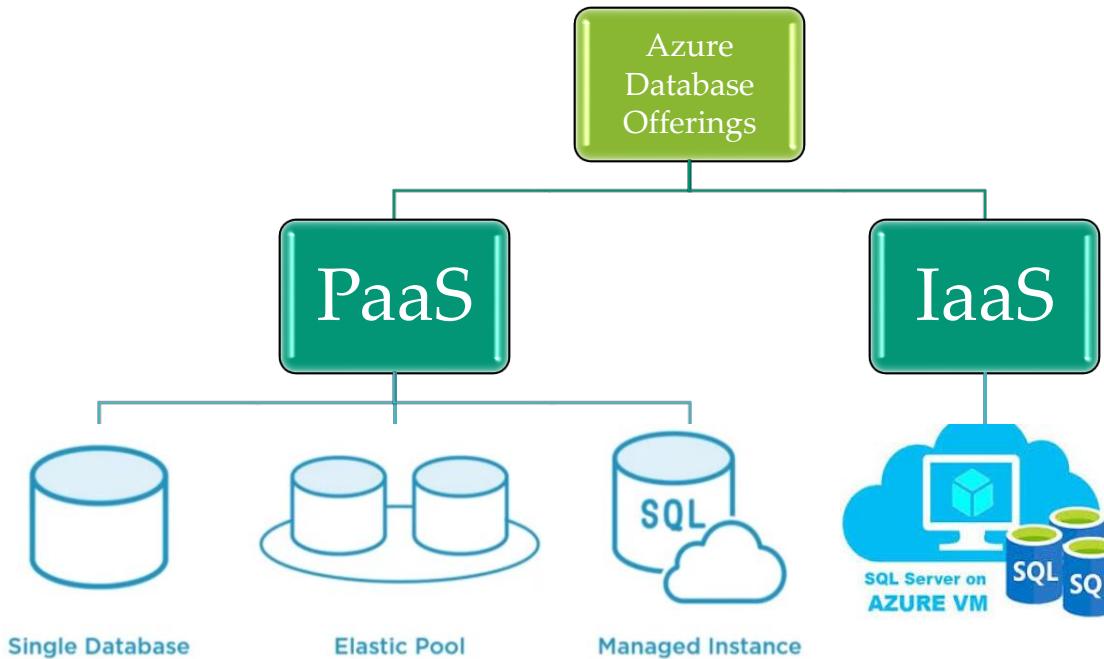
# Azure SQL Database Service Tiers



# Azure SQL Database Options



# Azure Database Elastic Pool



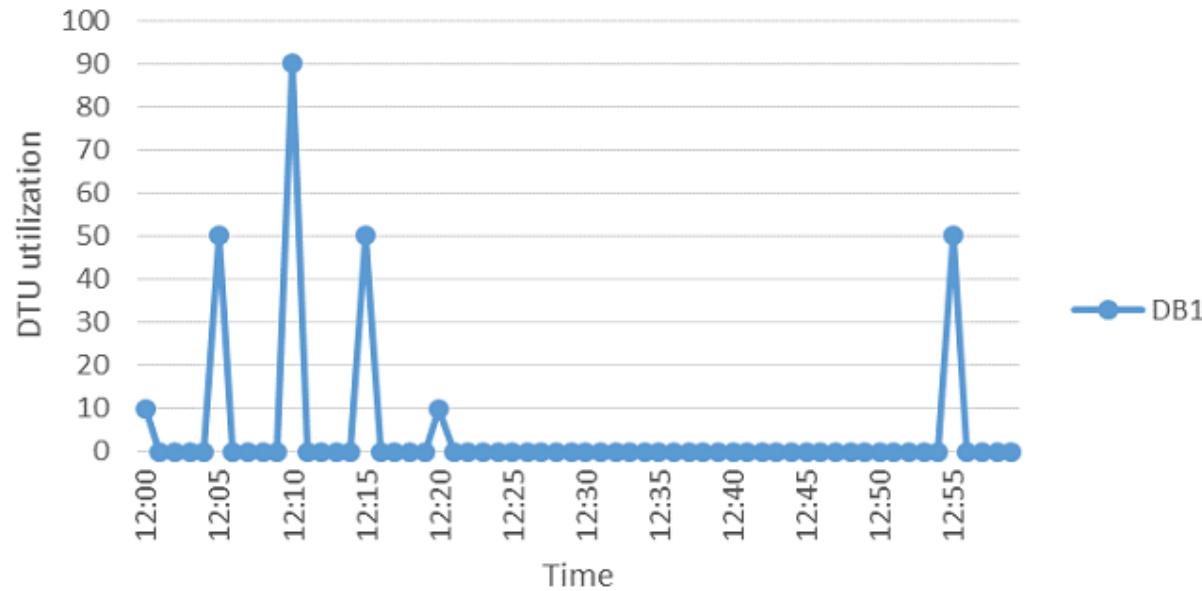
# SQL Database Elastic Pools

*An Azure SQL Elastic Pool allows you to allocate a shared set of compute resources to a collection of Azure SQL databases,*

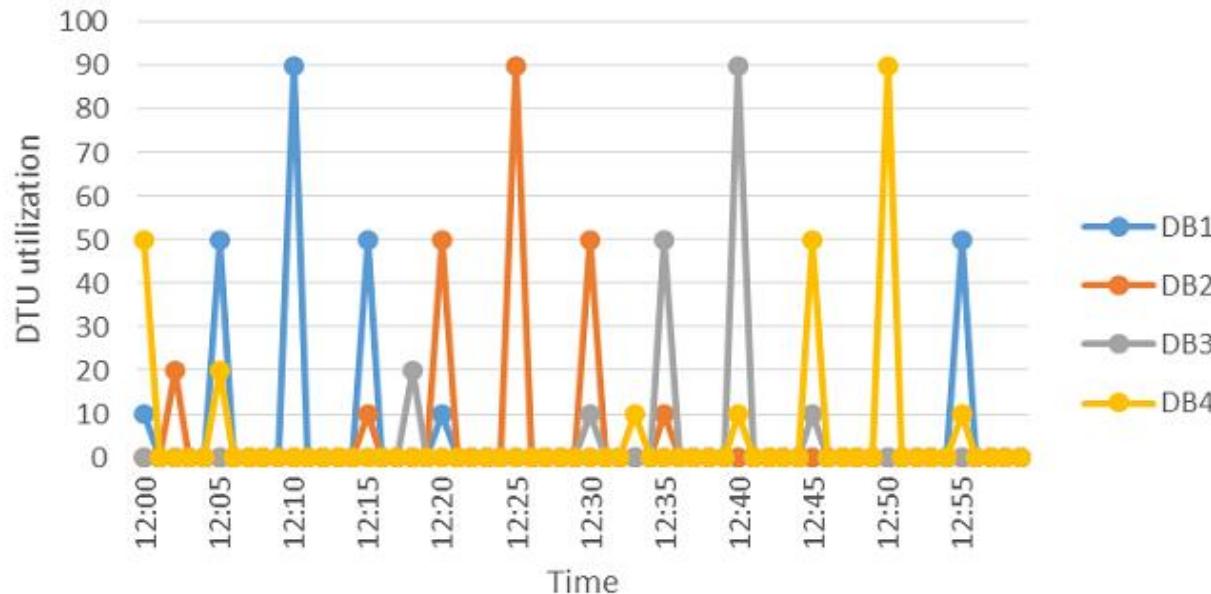
# Database Challenges

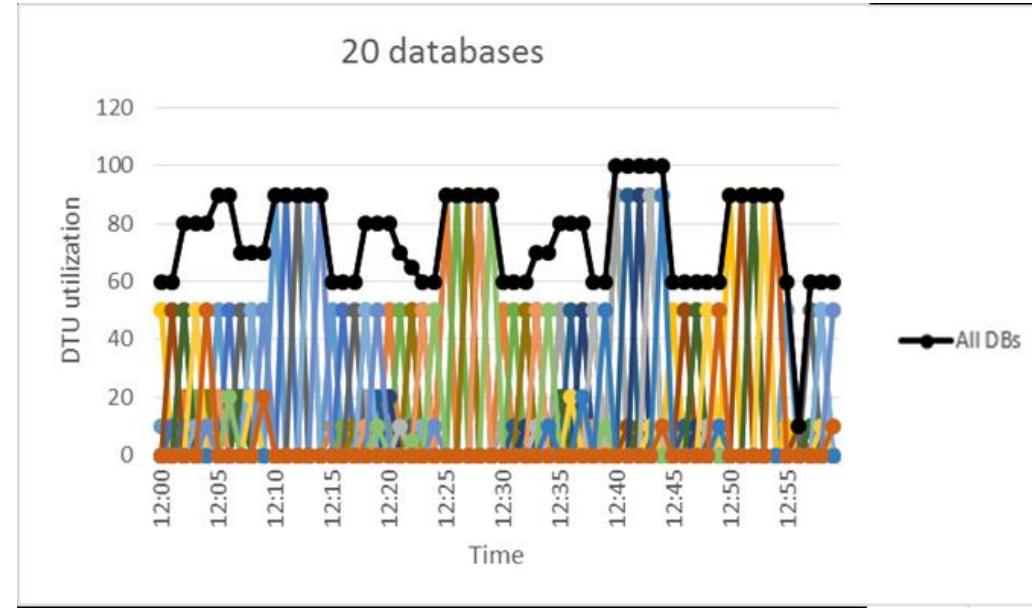


### 1 database

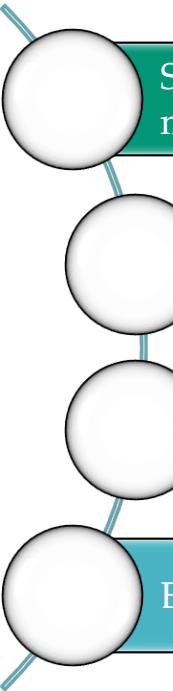


### 4 databases





# Azure SQL Database Elastic Pool



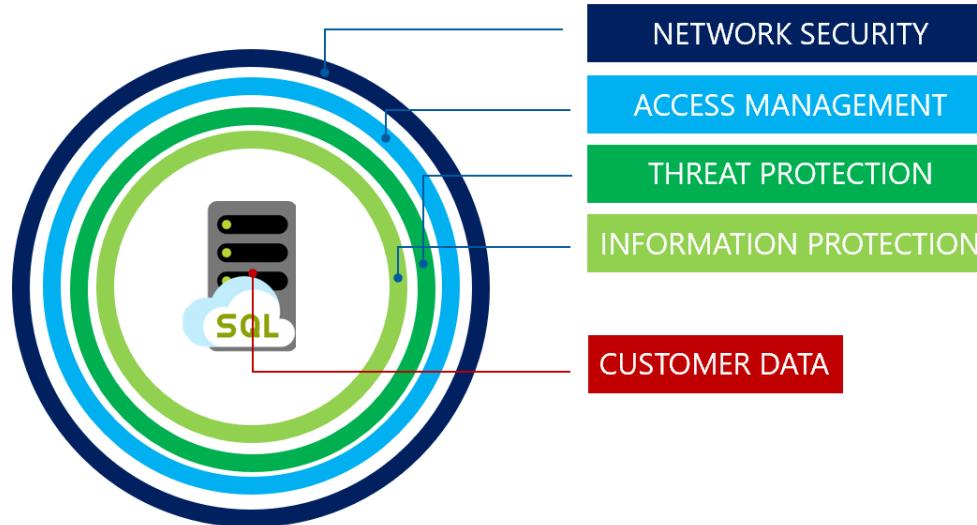
So Azure SQL elastic pool is a cost- effective solution for managing and scaling multiple databases that have varying and unpredictable usage demands.

The databases in an elastic pool are on a single Azure SQL Database server and share a set number of resources at a set price.

Elastic pool enables developers to optimize the price performance for a group of databases within a prescribed budget.

Elastic pools prevent over-provisioning or under-provisioning of resources.

# Security



# Network security

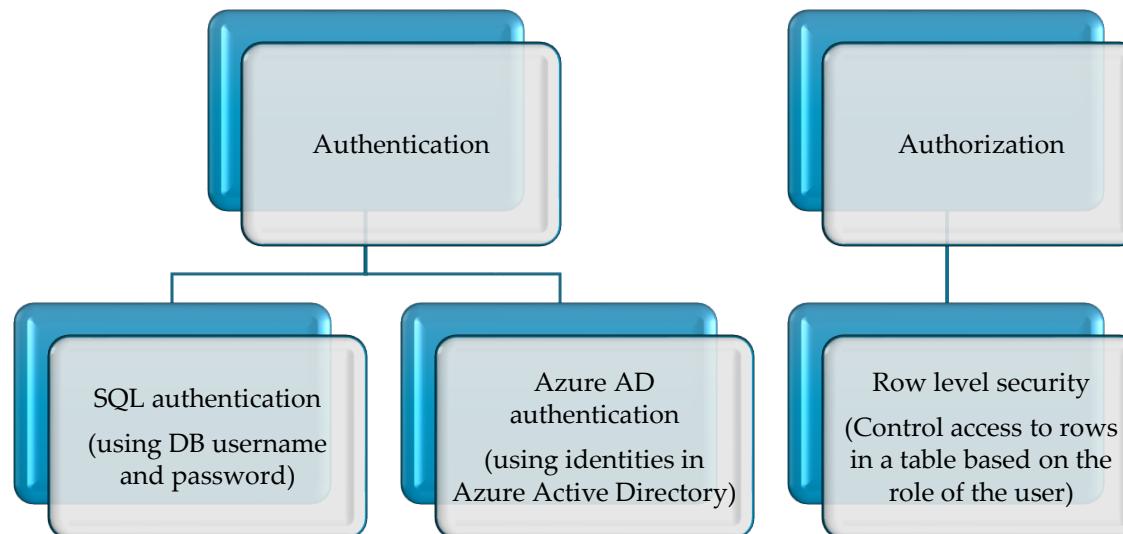
## IP firewall rules

- Grant access to databases based on the originating IP address of each request.

## Virtual network firewall rules

- Enable Azure SQL Database to only accept requests originating from subnets inside a virtual network.

# Access Management



# Threat Protection

Logs monitoring  
in Azure  
Monitor logs  
and Event  
Hub

Advanced  
Threat  
Protection

- Tracks database activities and helps maintaining compliance with security standards

- Analyzes your SQL Server logs to detect unusual behavior and potentially harmful attempts

# Information Protection

## Transport Layer Security TLS

- Always enforces encryption for all connections

## Transparent Data Encryption

- (Protects data at rest from offline access to raw files or backups)

## Dynamic Data masking

- (Protects sensitive data by masking it for non-privileged users)

# Information Protection

## Transport Layer Security TLS

- Always enforces encryption for all connections

## Transparent Data Encryption

- (Protects data at rest from offline access to raw files or backups)

## Dynamic Data masking

- (Protects sensitive data by masking it for non-privileged users)

# Security Management

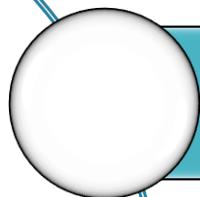
Vulnerability assessment

- Discover track and remediate potential database vulnerabilities.

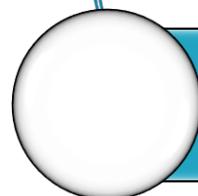
Data discovery & Classification

- Identify and label sensitive data for monitoring and alerting

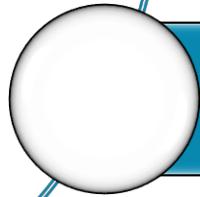
# Managed Instance Advance Security



Native virtual network implementation and connectivity to your on-premises environment using Azure Express Route or VPN Gateway.



In a default deployment, SQL endpoint is exposed only through a private IP address, allowing safe connectivity from private Azure or hybrid networks.



Single-tenant with dedicated underlying infrastructure (compute, storage).

# Azure SQL DB

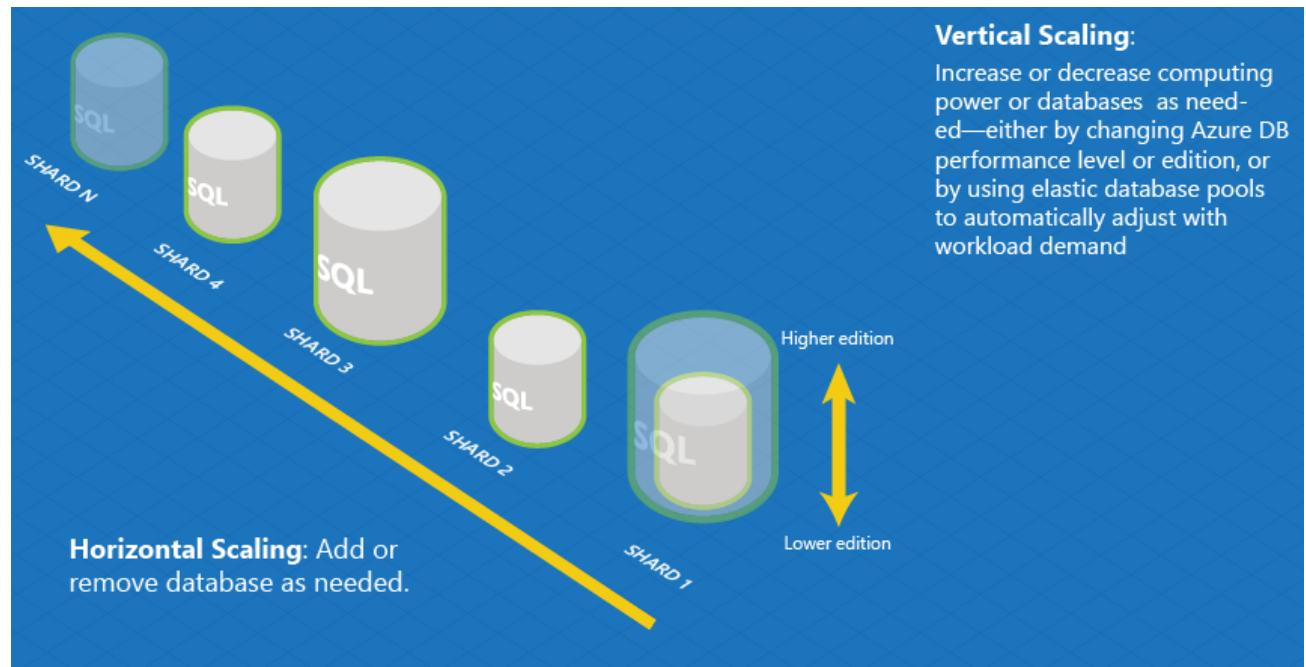
## Scaling

Prerequisite: Purchasing model and service tier

# Azure SQL Database Scaling

Azure SQL Database supports two types of scaling:

- **Vertical scaling:** Scale up or down the database by adding more compute power.
- **Horizontal scaling:** Scale out or Add more databases and to shard your data into multiple database nodes.



# Vertical vs Horizontal Scaling

Azure SQL Database supports two types of scaling:

- **Vertical scaling:**
  - Scale up or down the database by adding more compute power.
  - CPU Power, Memory, IO throughput, and storage
  - DTU and vCore models to scale
  - Dynamic Scalability (Note: this is not auto-scale)
  - Any change that you made will be almost instant.
- **Horizontal scaling:** Scale out or Add more databases and to shard your data into multiple database nodes.



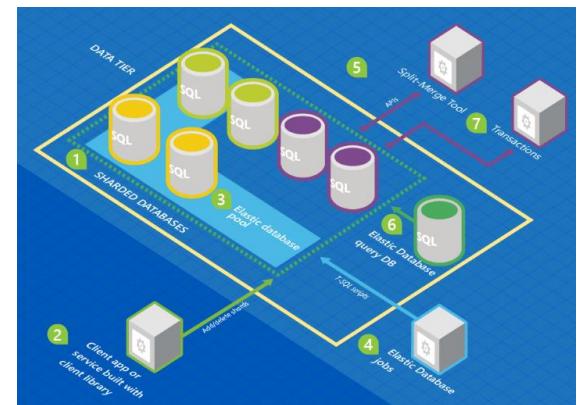
# Azure SQL Database Scaling

## Read Scale Out

- Allows you to use the capacity of the read-only replicas for read-only queries.
- Feature is intended for the applications that include logically separated read-only workloads, such as analytics
- Benefit: When Secondary nodes handles heavy reports and analytical queries, primary writable node saved resources that might be used to improve the performance
- Secondary nodes is asynchronous
- Premium ( DTU-based model ) or in the Business Critical ( vCore-based model)
- Also available in Hyperscale with secondary replica creation

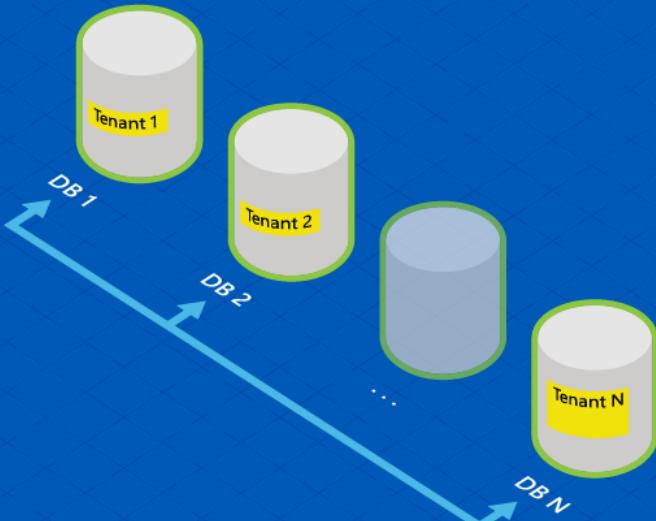
## Global Scale-out/Sharding

- Split your data into multiple database nodes.
- Every database shard is an independent database where you can add or remove resources as needed.
- Application may access only the shard associated to that region without affecting other shards.
- Why?
  - Data or transaction throughput exceed the capabilities of individual database
  - Tenants may require physical isolation
  - Different sections of a database may need to reside in different geographies for compliance, performance, or geopolitical reasons.



# Single Tenancy vs Multi Tenancy

Single Tenancy



Multi Tenancy



# Azure SQL Database Scaling

## Change Service Tier

- From Standard/General Purpose to Premium/Business Critical.
- In Standard/General Purpose - Data stored on Azure premium disks
- In Premium/Business Critical - Data stored on local SSD

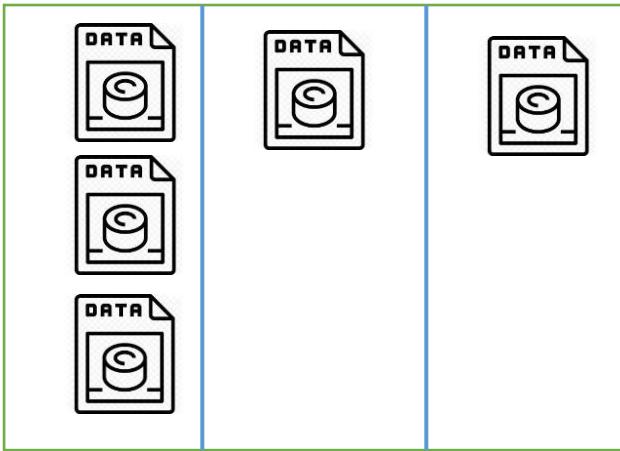


# Azure SQL DB

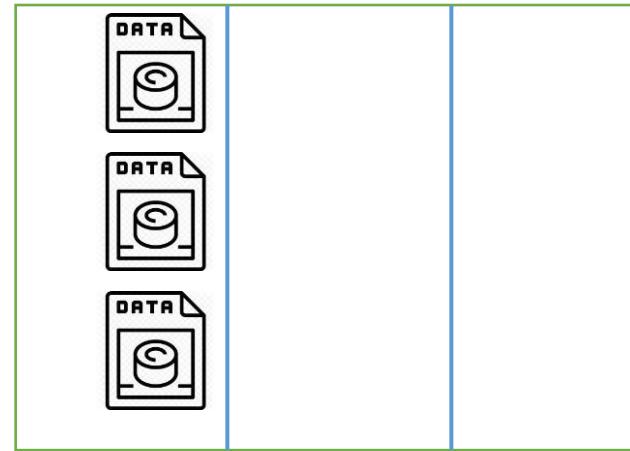
HA and DR Options

# Azure SQL DB – HA and DR Options

Region A



Region B (Read)



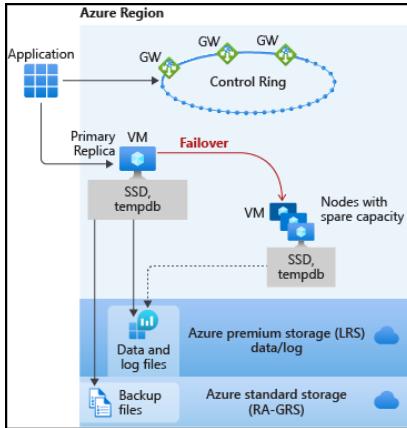
Storage Clusters

Each cluster is physically separate in what's called an availability zone, with its own separate utilities and networking.

Region B

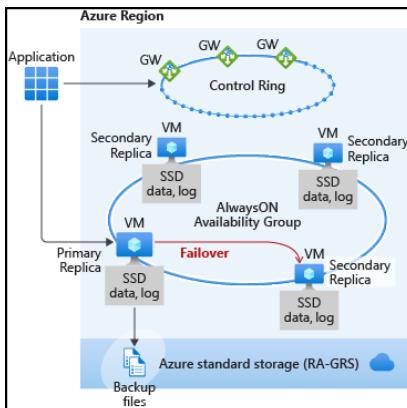
Hundreds of miles away from the primary region to prevent data loss in the event of a natural disaster.

# Azure SQL DB – High Availability Architecture



## Standard availability model

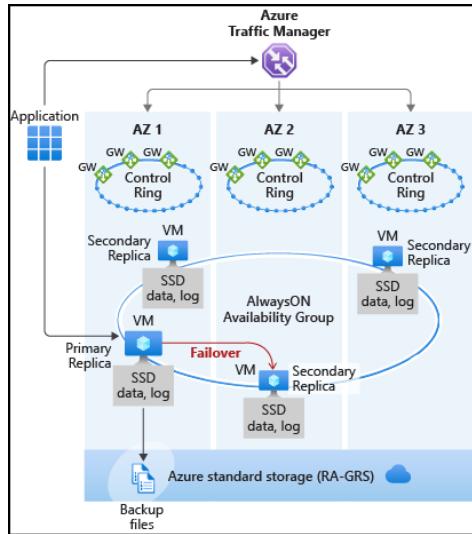
- Separation of compute and storage
- Performance degradation during maintenance activities
- Budget-oriented business applications
- Basic, Standard, and General Purpose service tier availability



## Premium availability model

- Integrates compute resources and storage (locally attached SSD) on a single node
- Replicating both compute and storage to additional nodes creating a three to four-node cluster.
- Data is synchronized to at least one secondary replica before committing each transaction.
- Targets mission critical applications with high IO performance and high transaction rate
- Guarantees minimal performance impact during maintenance activities.
- Premium and Business Critical service tier availability
- Read Scale-Out feature

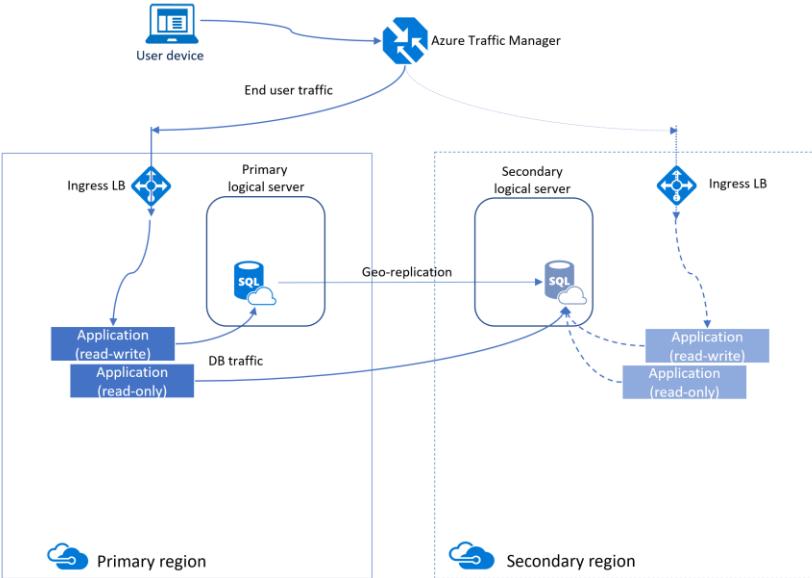
# Azure SQL DB –Zone Redundancy



## Zone redundant configuration

- Place different replicas of the Business Critical database to different availability zones in the same region
- Does not create additional database redundancy
- Enable it at no extra cost
- Supported in the Premium and Business Critical service tiers
- Not available in SQL Managed Instance.
- Increased network latency may increase the commit time and thus impact the performance of some OLTP workloads.

# Azure SQL DB – Geo Replication



- **Issues:**
  - Supports only manual failover
  - End-point connection must be changed in the application after the failover
  - Must have the same firewall rules and the logins to run applications successfully without any discrepancies
- **Create a readable secondary database in the same region or cross-region**
- **Use cases:**
  - Can failover to the secondary database in case of an outage
  - Migrate a database from one server to another server in the same or cross region with minimal downtime.
- We can create up to four secondaries for each primary database.
- **Data Loss:**
  - Uses the Always-on feature to replicate committed transactions to the secondary database asynchronously.
  - May lag the primary database at any point in time.
- **Manual - Forced Failover**
  - This will make your secondary database immediately online and start accepting connections. Forced failover may result in data loss.

# Compare geo-replication with failover groups

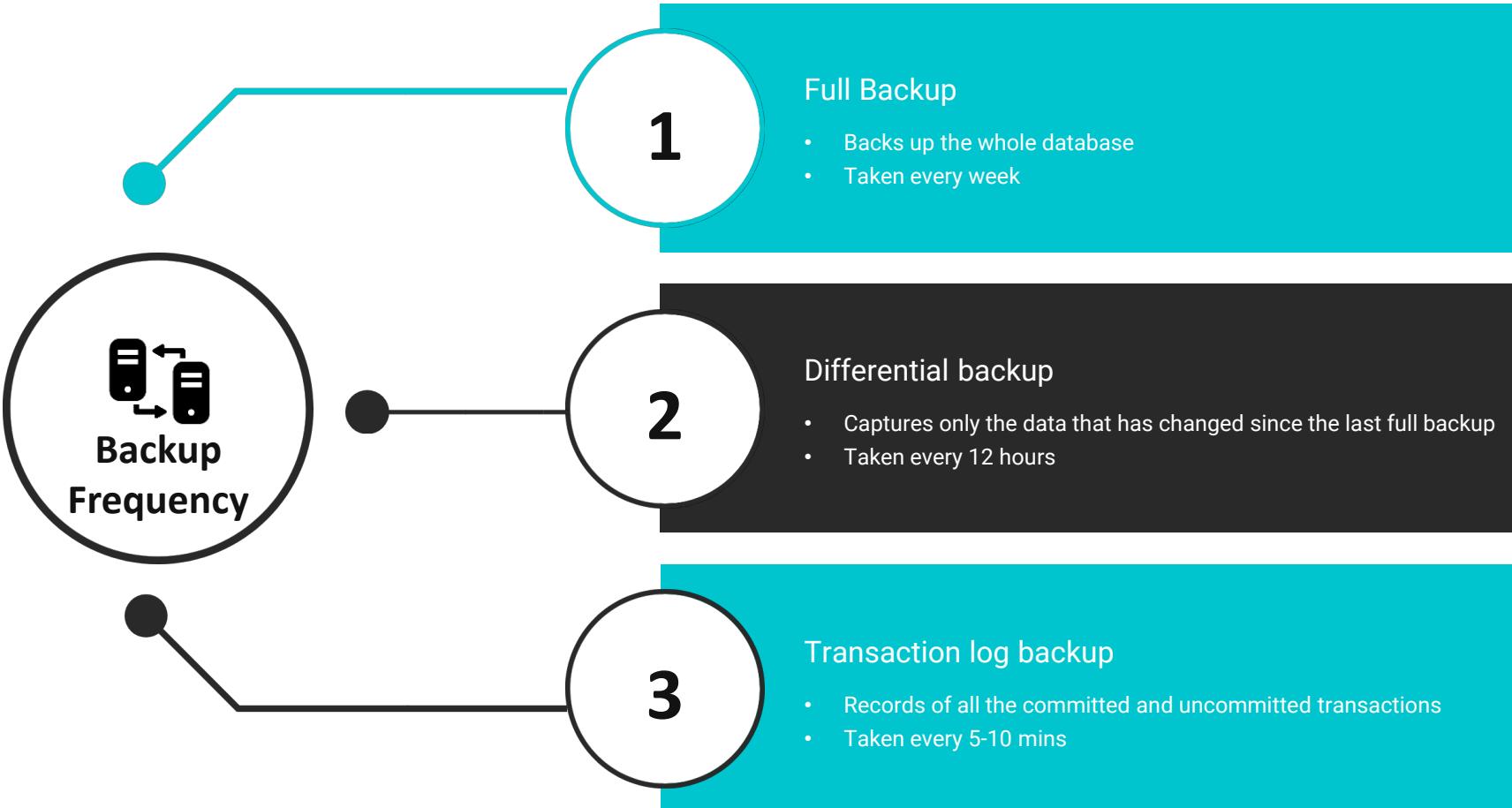
[Auto-failover groups](#) simplify the deployment and usage of [geo-replication](#) and add the additional capabilities as described in the following table:

	<b>Geo-replication</b>	<b>Failover groups</b>
<b>Automatic failover</b>	No	Yes
<b>Fail over multiple databases simultaneously</b>	No	Yes
<b>User must update connection string after failover</b>	Yes	No
<b>SQL Managed Instance support</b>	No	Yes
<b>Can be in same region as primary</b>	Yes	No
<b>Multiple replicas</b>	Yes	No
<b>Supports read-scale</b>	Yes	Yes

# Azure SQL DB

## Backup and Restore

Protect your data from corruption or deletion



## Storage cost and Security of Backup files



### Storage Cost

Backup files are copied to RA-GRS standard blob storage by default to paired region



### Security

Backup are automatically encrypted at rest using TDE, blob storage is also protected

# Backup Retention Period



## Backup storage redundancy

- Point in time restore (PITR) - 7-35 days
- Long-term retention - Up to 10 years

## Long term retention (LTR)

- One or more long term retention periods to your database to meet regulatory, compliance or other business purposes
- Full backups can be taken up to 10 years
- Stored in RA-GRS blob storage
- Any change of the LTR policy applies to the future backups

## LTR and Managed Instance

- LTR is not yet available for databases in Managed Instances
- You can use SQL Agent jobs to schedule copy only database backups as an alternative to LTR beyond 35 days
- These backups can be kept in the Azure blob storage

# Backup Restore

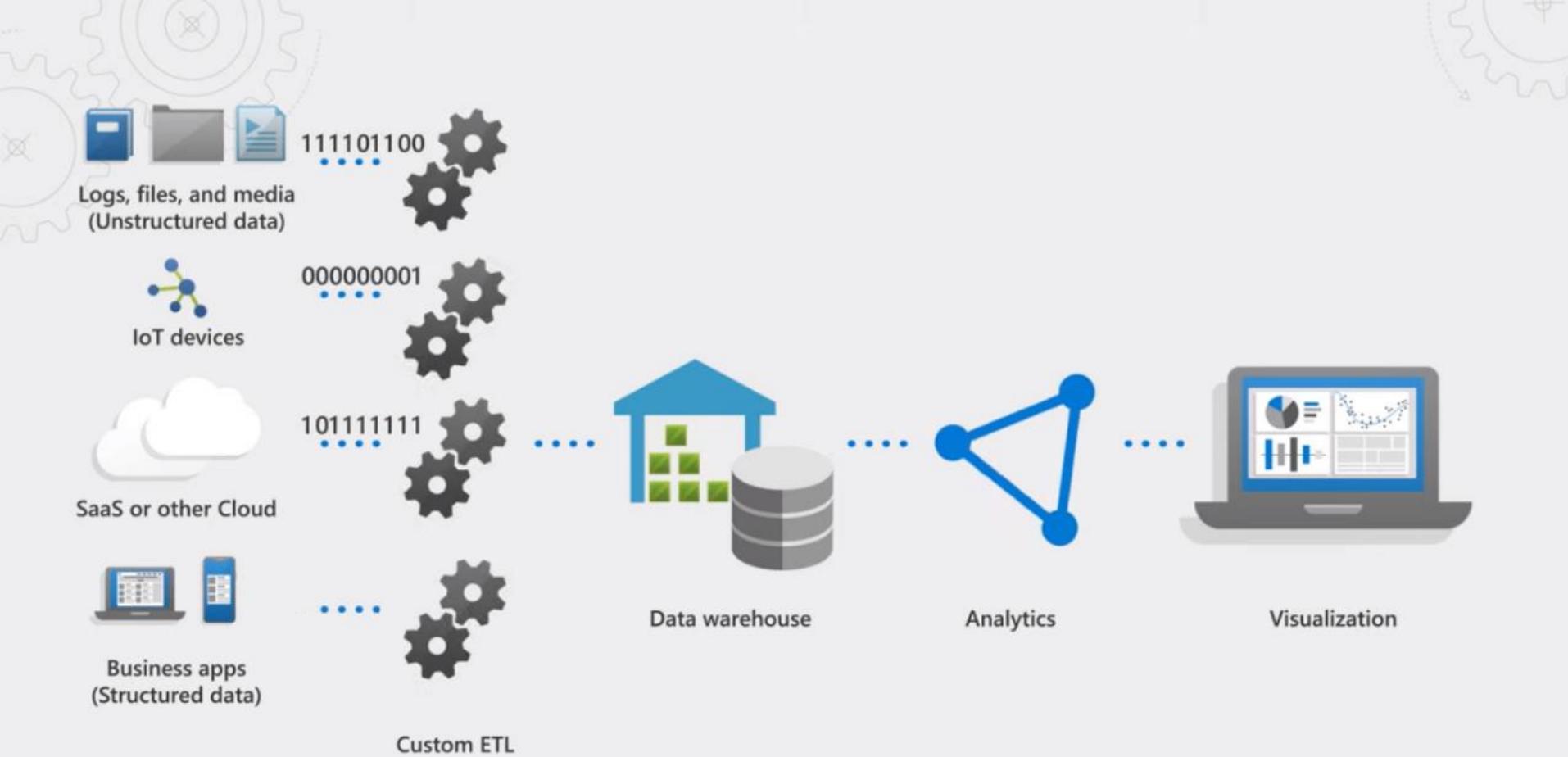


## Backup usage

- Point-in-time restore of existing database
- Point-in-time restore of deleted database
- Geo-restore
- Restore from long-term backup
- *If you delete an Azure Logical SQL Server, all elastic pools and databases that belong to that logical server are also deleted and cannot be restored*

## Restore Time is impacted By

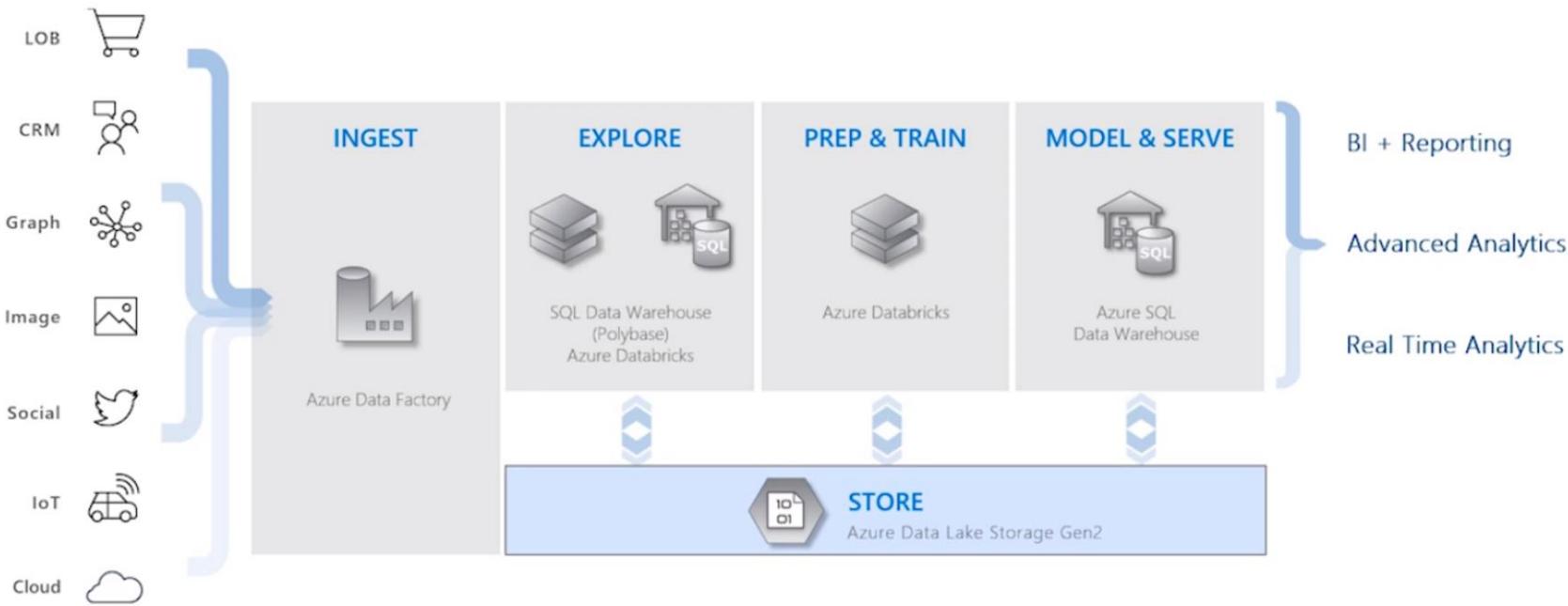
- Size of database & Compute size of the database
- Number of transaction logs and Amount of activity
- Network bandwidth if the restore is to a different region
- Concurrent restore requests being processed region



## Old data warehouse

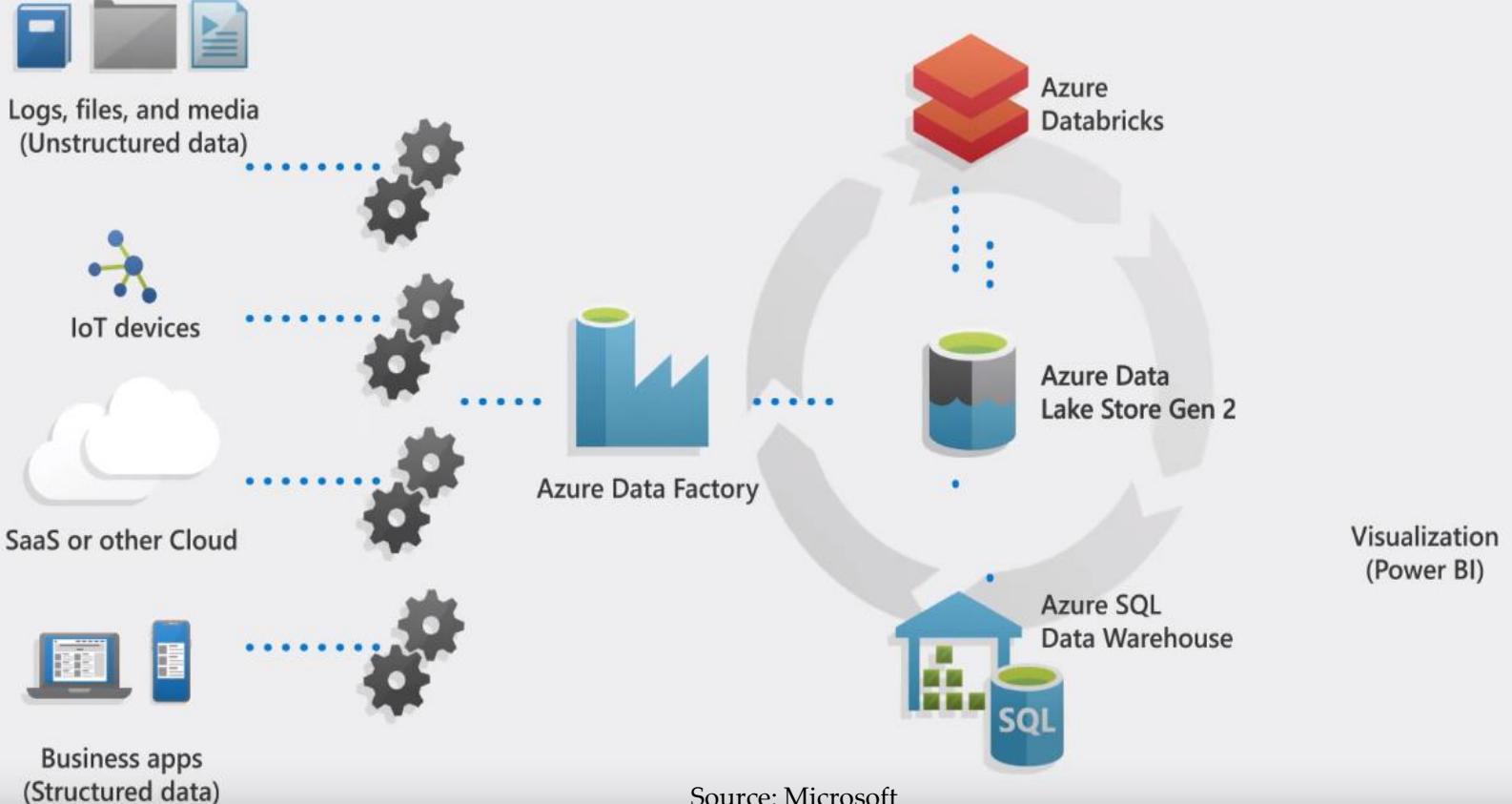
Source: Microsoft

# Modern Data Warehousing

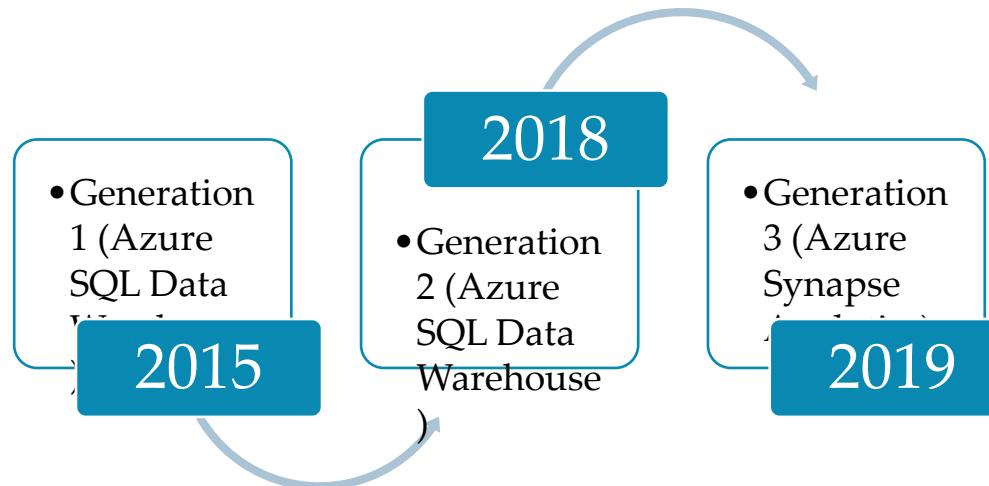


Source: Microsoft

# Modern Data Warehouse



# Azure SQL Data warehouse



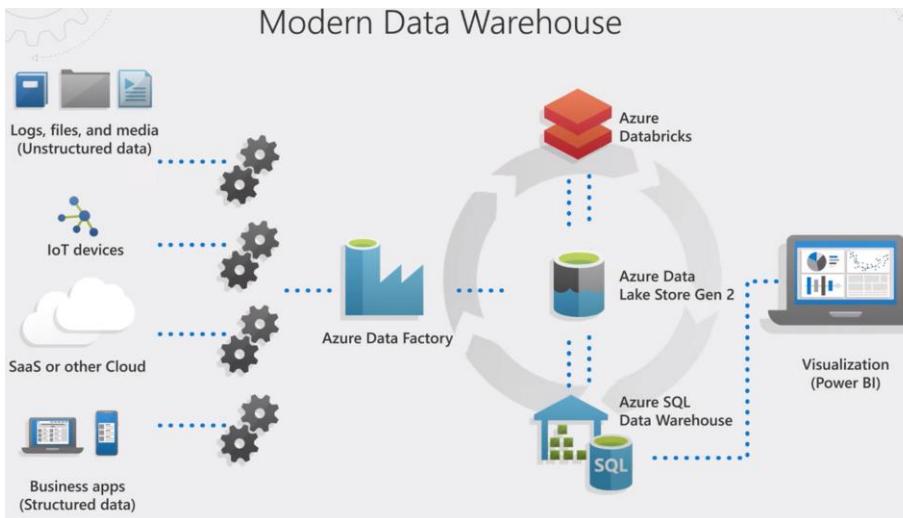
# Azure Synapse Analytics

*"Synapse is the next generation of Azure SQL Data Warehouse, blending big data analytics, data warehousing, and data integration into a single unified service that provides end-to-end analytics with limitless scale."*

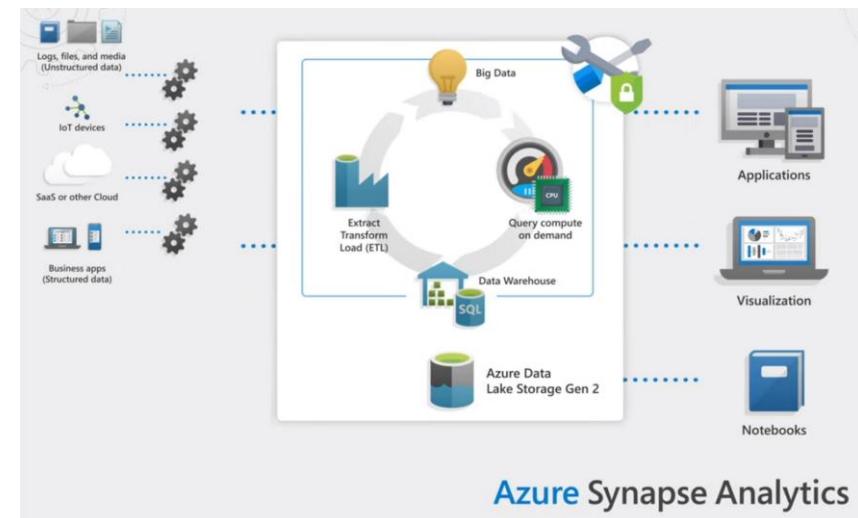
# Azure Synapse Analytics



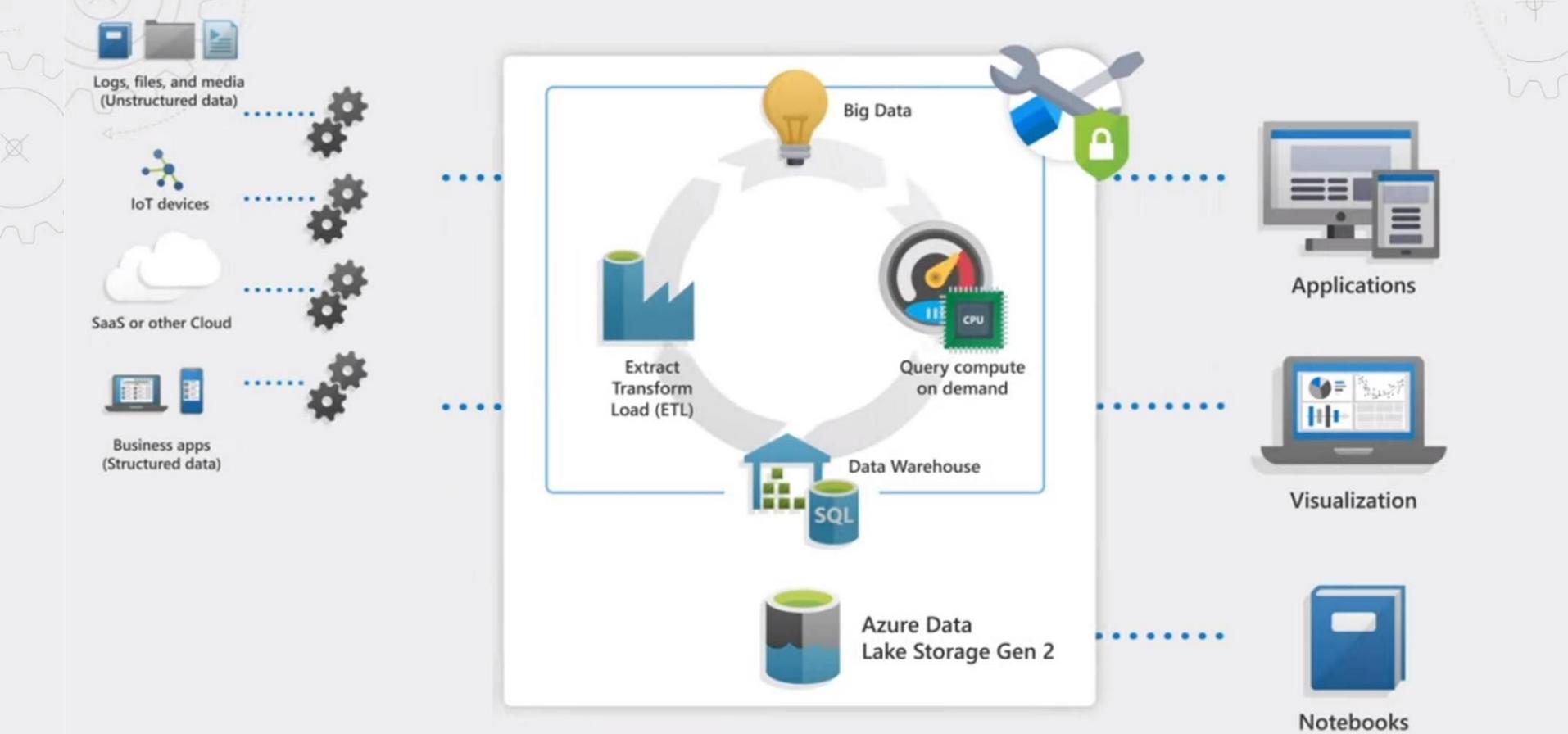
# Modern vs Synapse Architecture



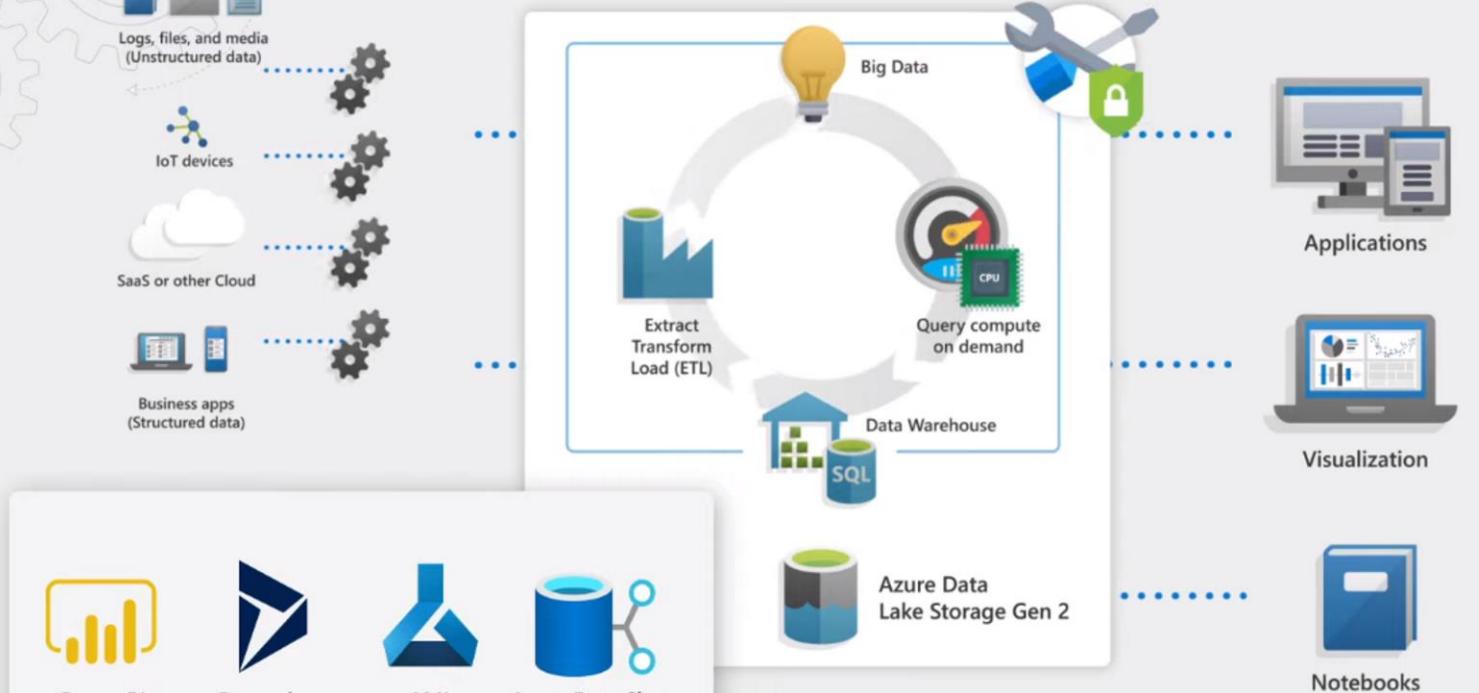
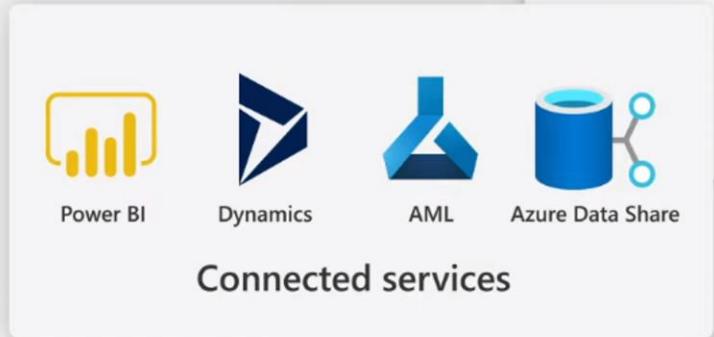
Modern Data Warehouse Architecture



Synapse Analytics Architecture



# Azure Synapse Analytics



## Azure Synapse Analytics

# Azure Synapse Analytics



# Demo – Provision Azure Synapse Service

1. Create SQL Server
2. Create Synapse SQL Pool (Azure SQL Data Warehouse)
3. Pause/Resume Compute Node
4. Create Firewall Rule
5. Connect with Microsoft SQL Server Management Studio



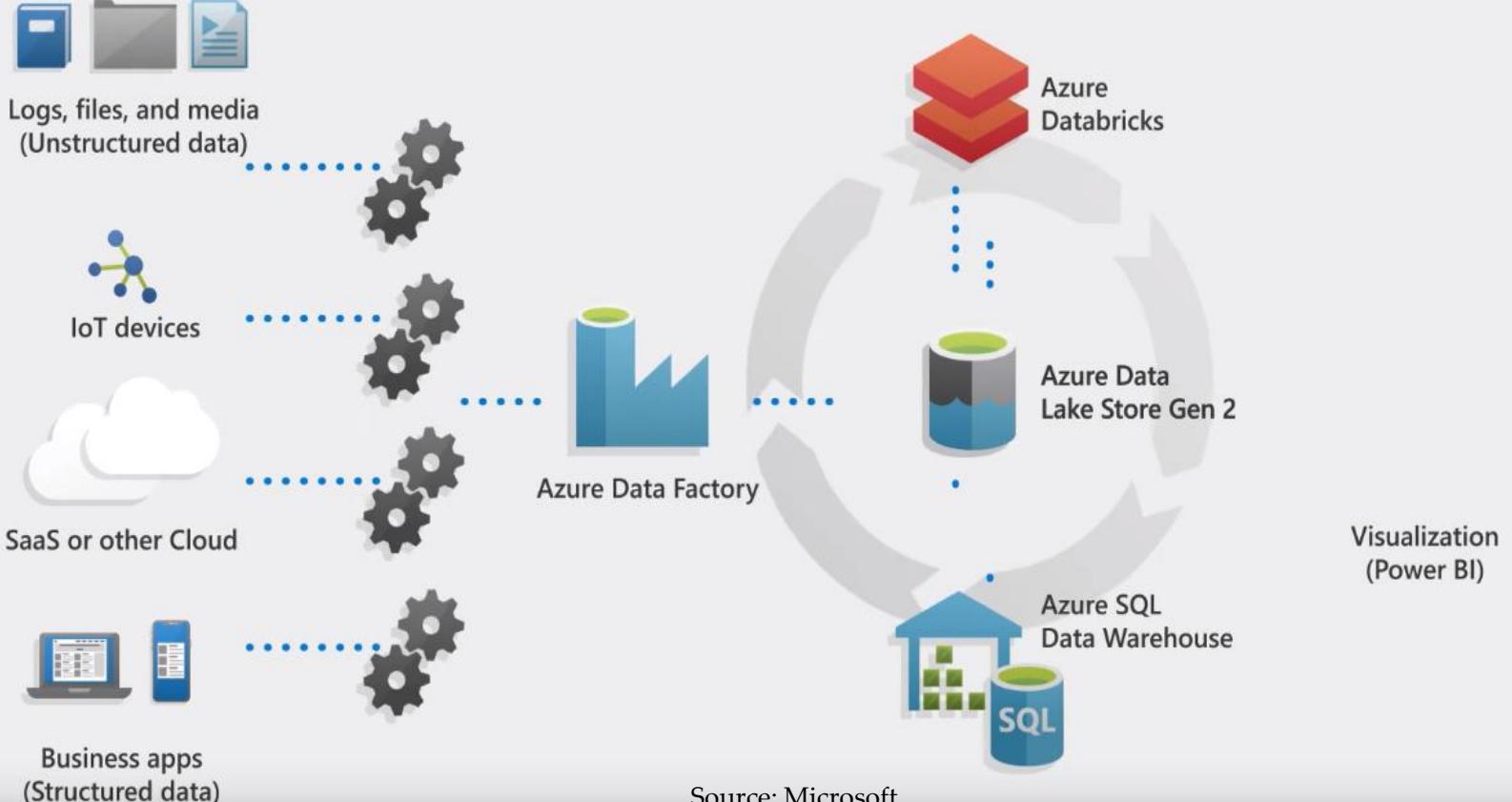
Azure  
Synapse  
Analytics

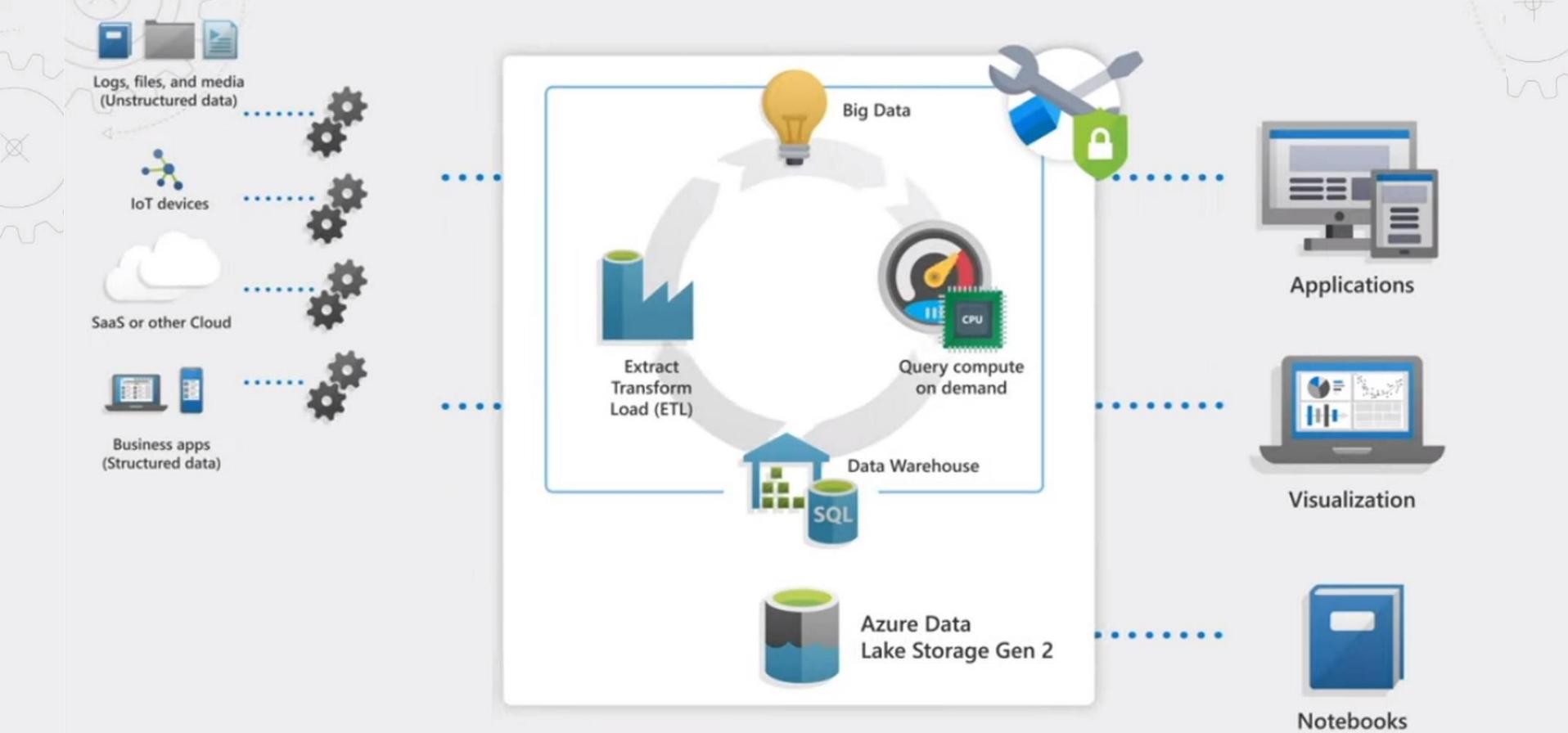


Azure  
Synapse  
Analytics

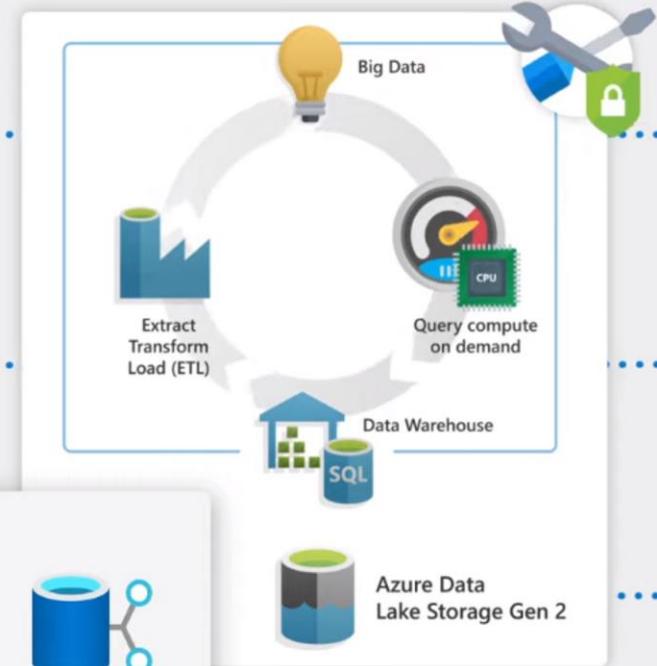
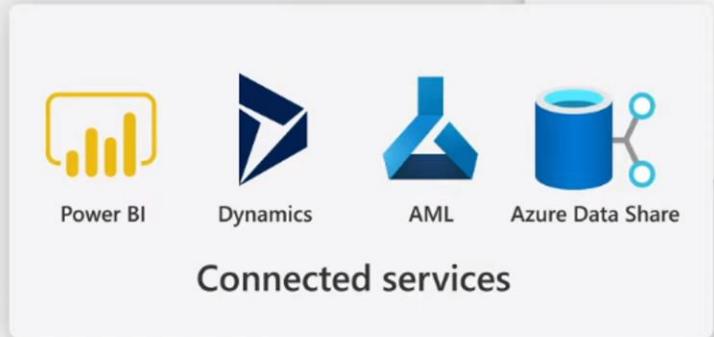


# Modern Data Warehouse





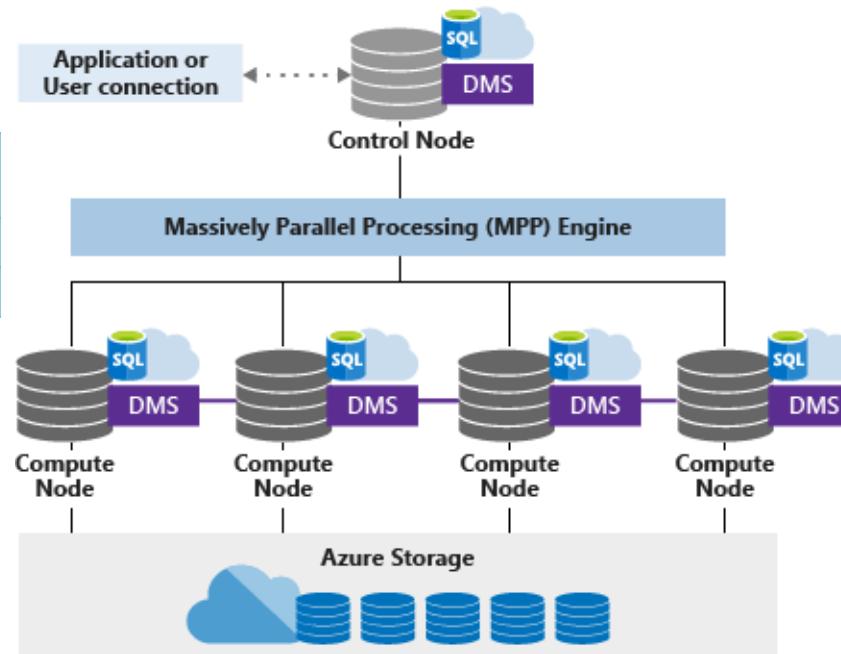
# Azure Synapse Analytics



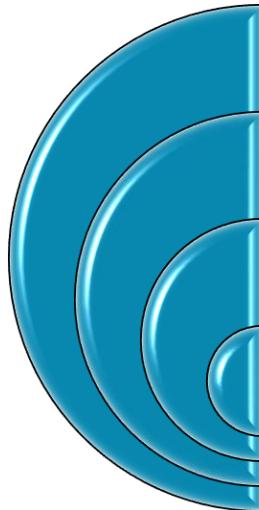
# Azure Synapse Analytics

# Azure Synapse MPP Architecture

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



# Azure Storage and Distribution



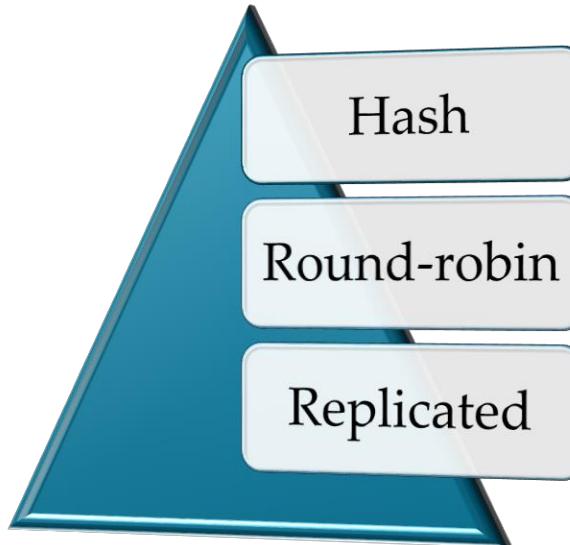
SQL DW charges separately for storage consumption

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 distribution

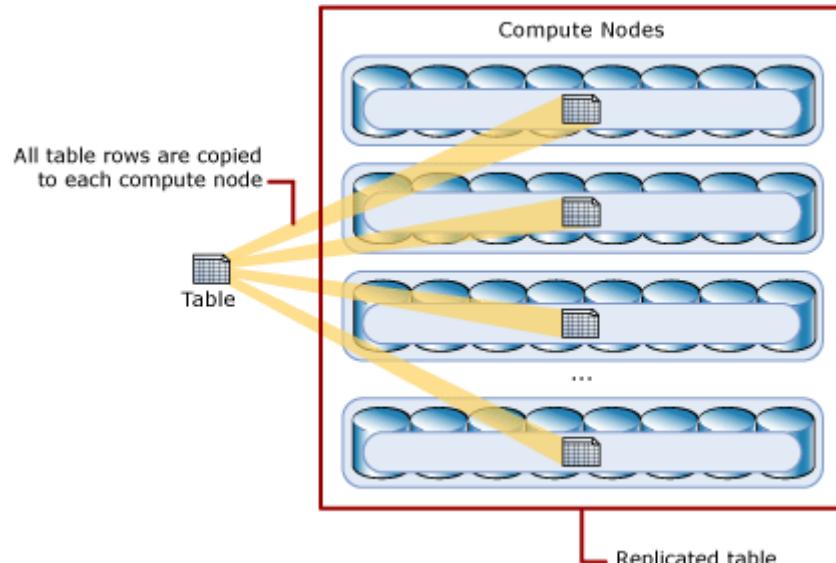
# Sharding Patterns



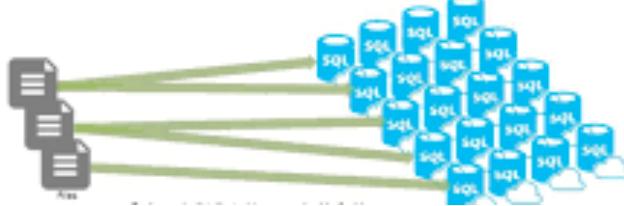
# Replicated Tables

- 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



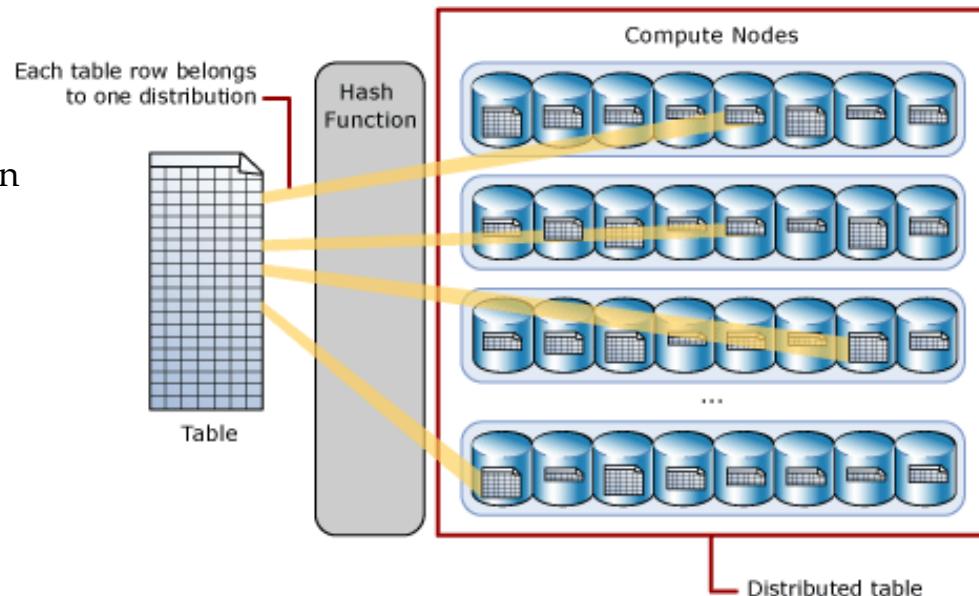
- Generally use to load staging tables
- Distribute data evenly across the table without additional optimization
- Joins are slow, because it requires to reshuffle data
- Default distribution type

```
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
)
;
```

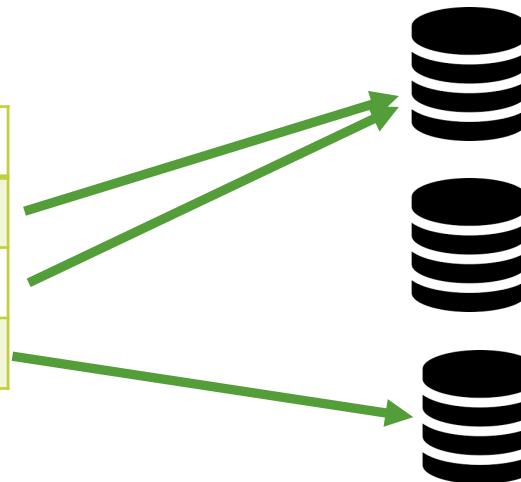
# Hash Distribution Tables

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



# Hash Distribution Tables

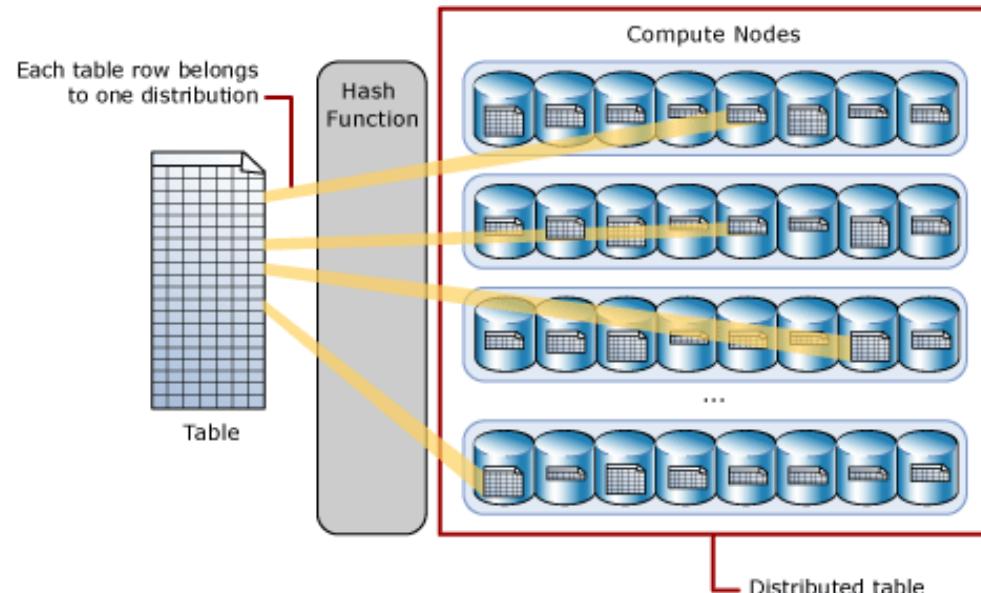
Record	Product	Store
1	Soccer	New York
2	Soccer	Los Angeles
3	Football	Phoenix



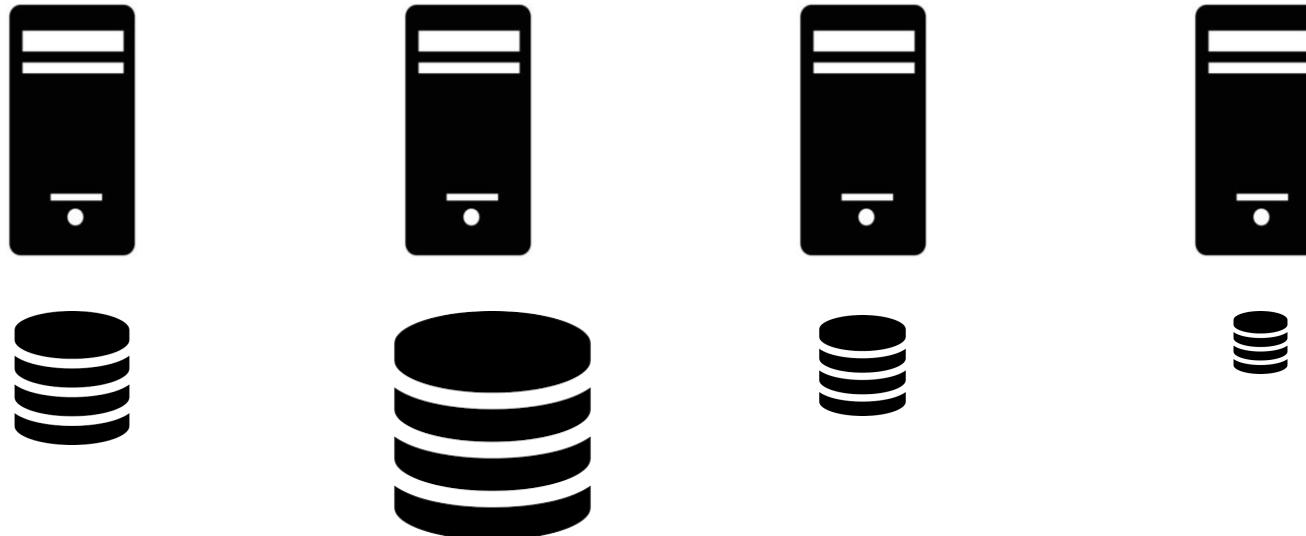
# Hash Distribution Tables

- Highest performance for large tables
- Each row belongs 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

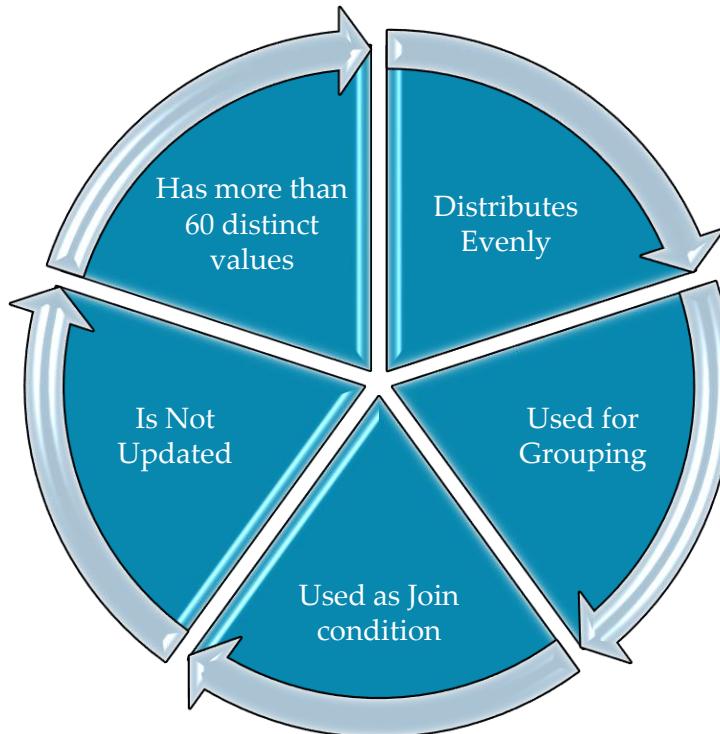


# Distribution key

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.

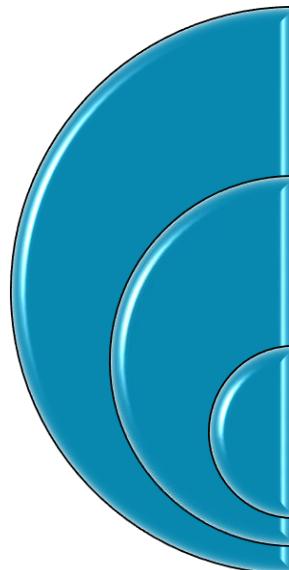
# Good Hash Key



# 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"><li>Many write transaction are on the table (insert/update/delete)</li><li>You change DWU provisioning frequently</li><li>You use only 2-3 columns, but your table has many columns</li><li>You index a replicated table</li></ul>
Round-robin (default)	<ul style="list-style-type: none"><li>Temporary/Staging table</li><li>No obvious joining key or good candidate column.</li></ul>	Performance is slow due to data movement
hash	<ul style="list-style-type: none"><li>Fact tables</li><li>Large dimension tables</li></ul>	The distribution key can't be updated

# Data types



Use the smallest data type which will support your data

Avoid defining all character columns to a large default length

Define columns as VARCHAR rather than NVARCHAR if you don't need Unicode

# 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
- Data is not in any particular order.

- Use when data has no natural order.

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

Heap

Clustered  
Index

Default table type

High compression  
ratio

Clustered  
columnstore

Ideally segments of  
1M rows

No Secondary  
Indexes

No index on the  
data

Fast Load

## Heap

No compression

Allows secondary  
indexes

Sorted index on  
the data

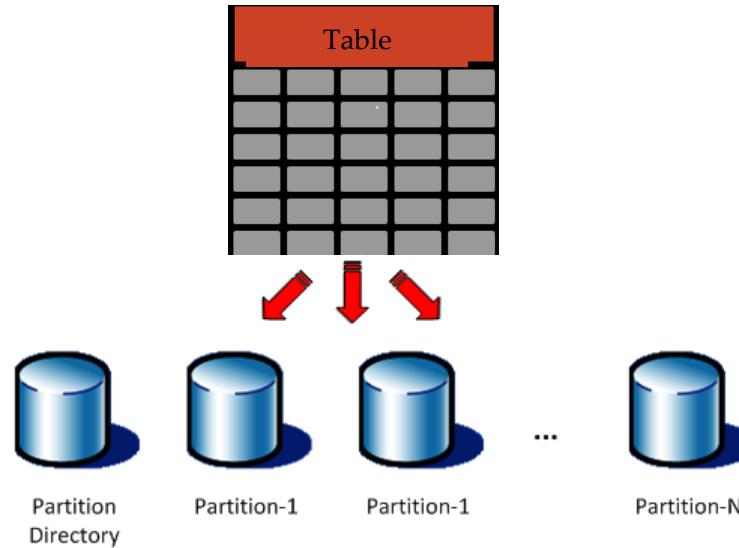
Fast singleton  
lookup

## Clustered B-Tree

No compression

Allows secondary  
indexes

# Table Partitioning



# Partitioning



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

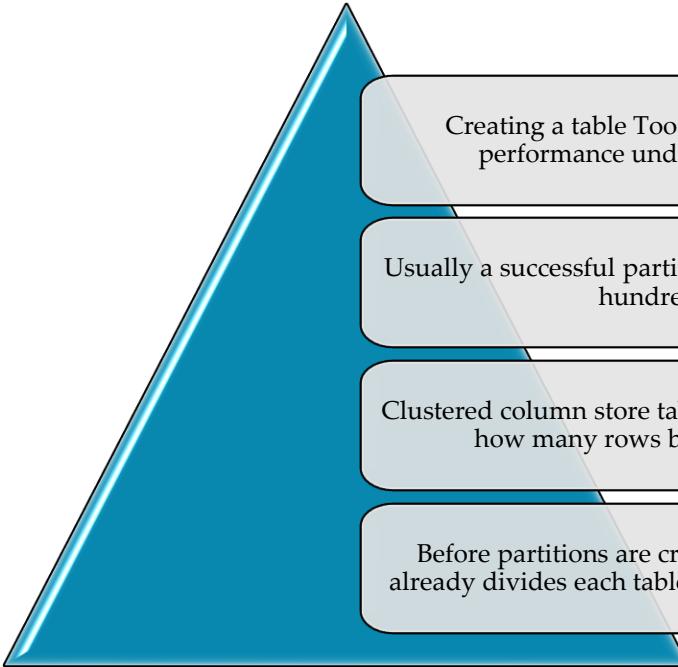
# Why Partitioning?

Easy load or  
unload data

Easy  
maintenance,  
rebuids or  
reorganizes

Improve Query  
Performance

# Partitions best practices



Creating a table Too many partitions can hurt performance under some circumstances

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

Clustered column store 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



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)



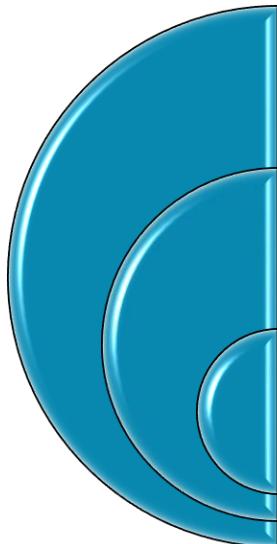
21 900 000 000 Rows



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

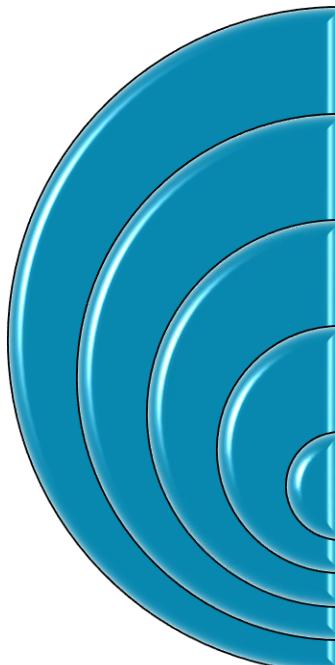


Large ones are better as Columnstores

Distributed through Hash key as much  
as possible as long as it is even

Partitioned only if the table is large  
enough to fill up each segment

# 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

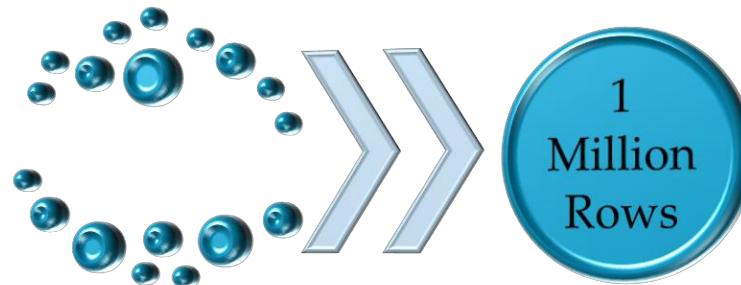
# Best Practices for Data Load

# Data Warehouse Readers

Your DWUs have a direct impact on how fast you can load data in parallel

# Optimize Insert Batch Size

Avoid trickle insert pattern. Ideal batch size is 1 million or more direct or in a file

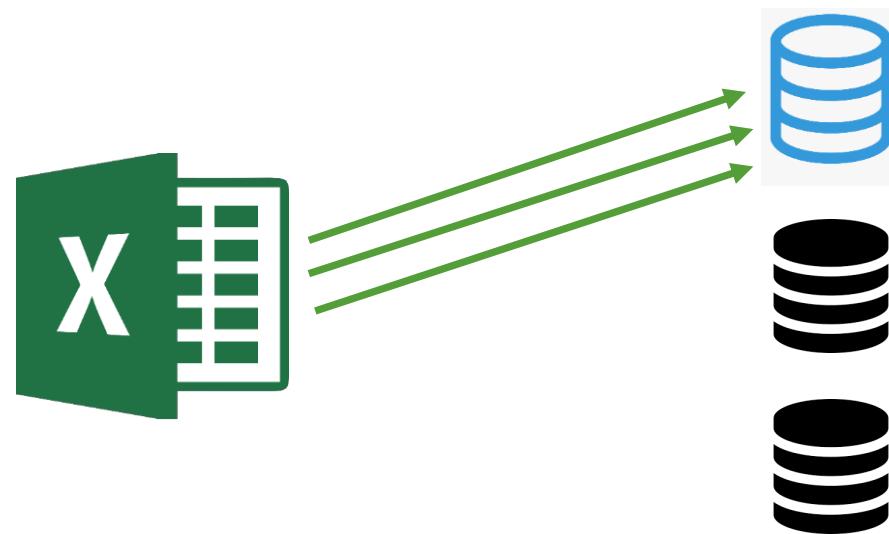


Combine rows to  
make it a batch of  
1 million

Ideal batch size

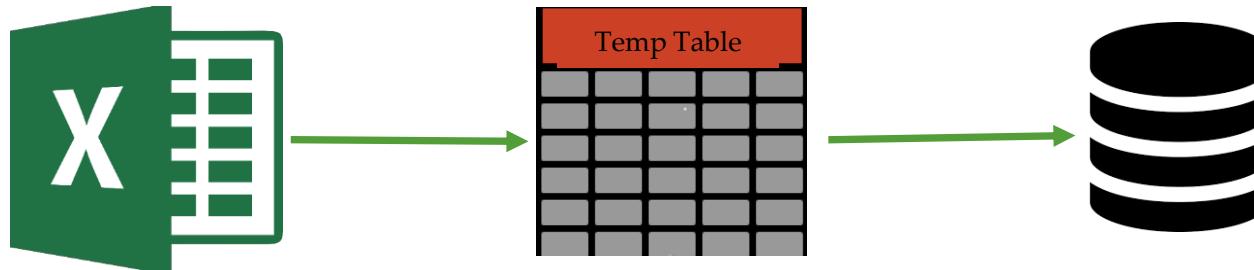
# Avoid ordered data

Data ordered by distribution key can introduce hot spots that slow down the load operation



# Using temporary tables

Stage and transform on a Temp Heap table before moving to permanent storage.



# CREATE TABLE AS

```
CREATE TABLE #tmp_fct
WITH
(
DISTRIBUTION = ROUND_ROBIN
)
AS
SELECT *
FROM
[dbo].[FactInternetSales];
```

- Fully Parallel operation
- It is minimally logged
- It can change: distribution, table type, partitioning

# Loading Methods

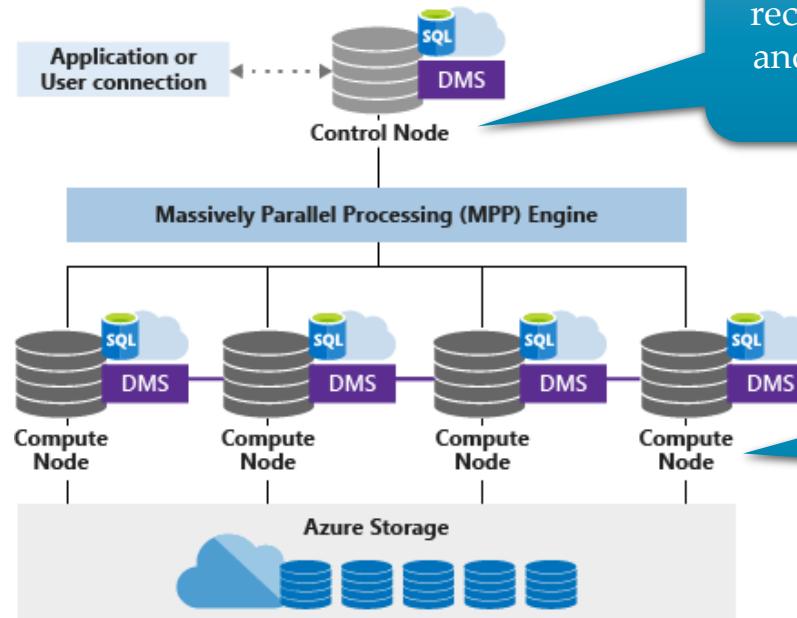
## Single client loading methods

- SSIS
- Azure Data Factory
- BCP
- Can add some parallel capabilities but are bottlenecked at the control node

## Parallel readers loading methods

- PolyBase
- Reads from Azure blob Storage and loads the contents into Azure SQL DW
- Bypasses the Control Node and loads directly into the Compute Nodes

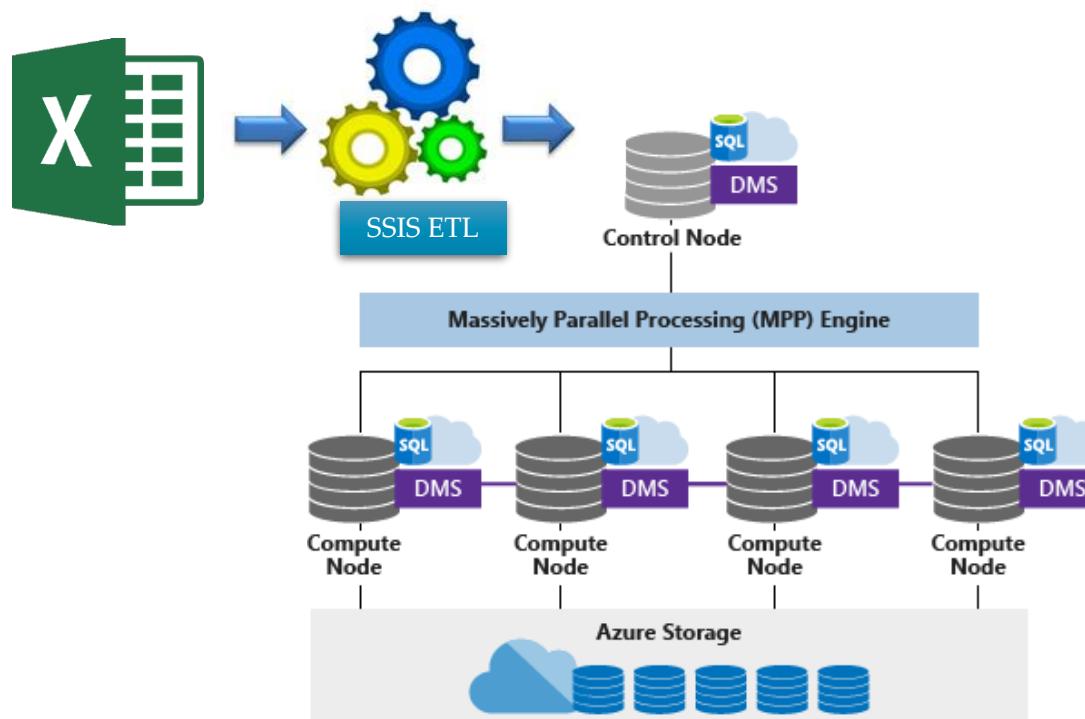
# Control Node



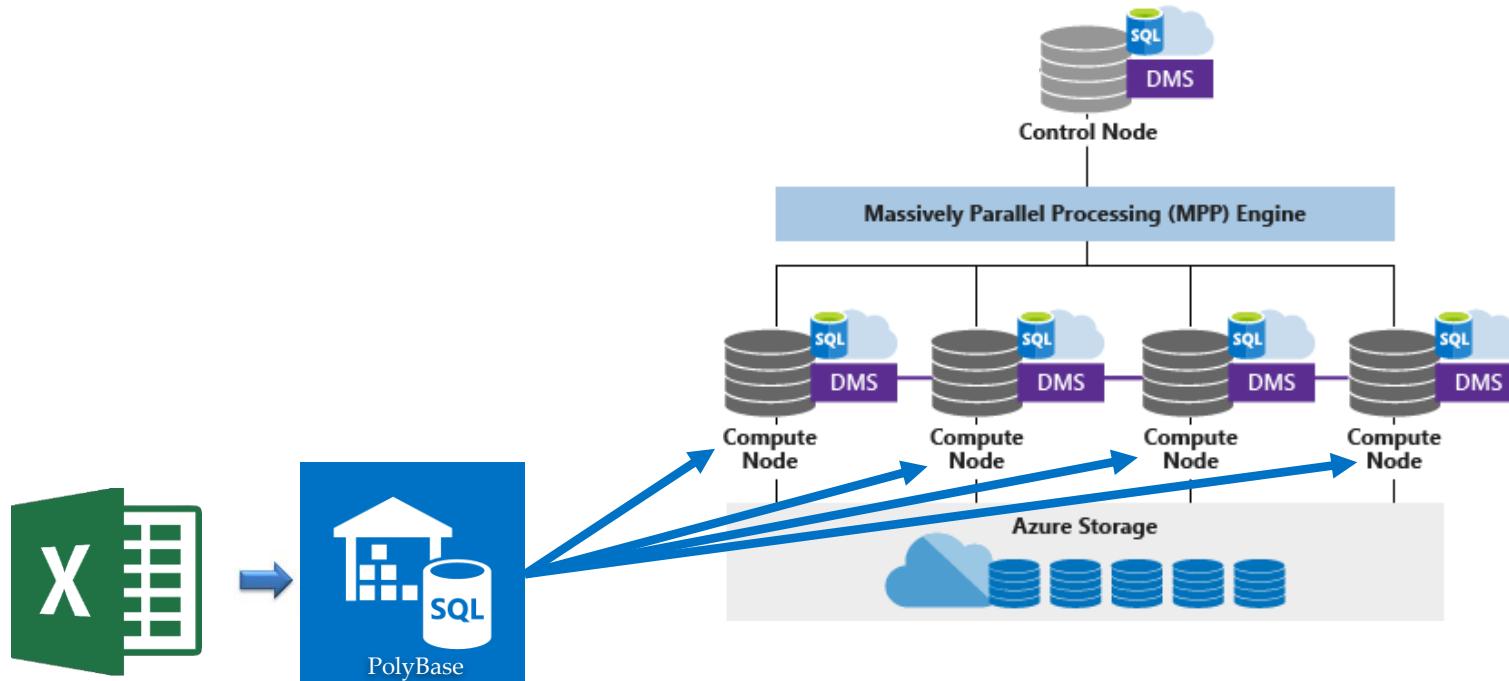
The **Control** node receives connections and orchestrates the queries

The **Compute** nodes do processing on the data and scale with the DWUs.

# Loading with SSIS



# Loading with PolyBase



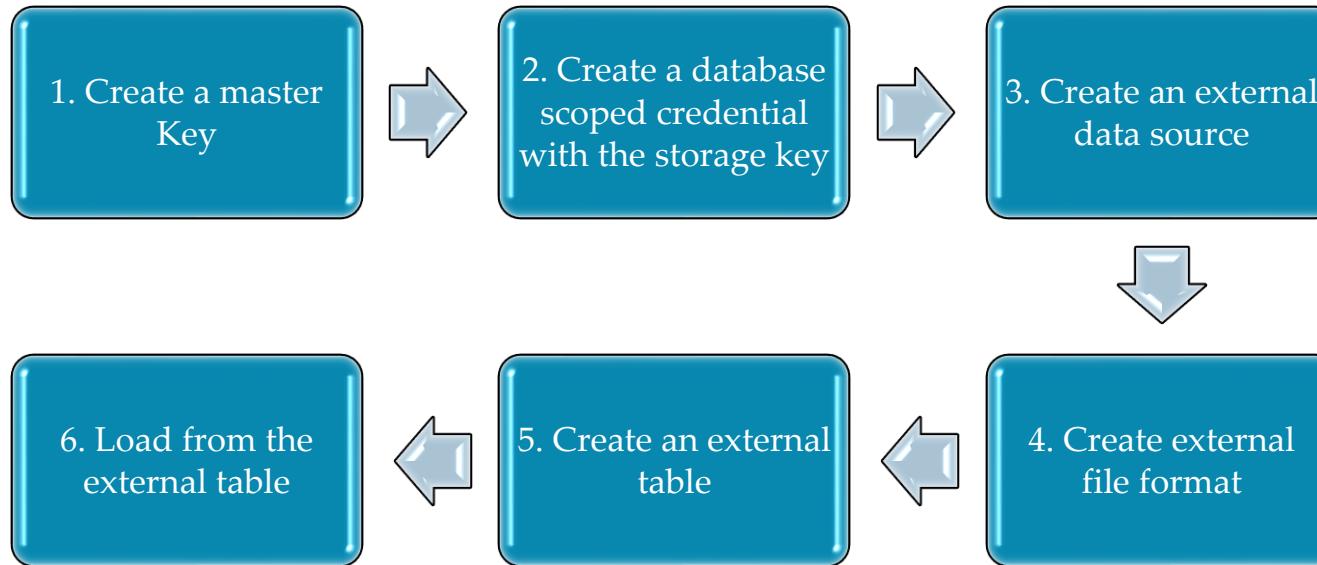


PolyBase can load data from UTF-8 delimited text files and popular Hadoop file formats like RC File, ORC, and Parquet



Multiple readers will not work against a  
compressed file

# PolyBase Setup



# DEMO

Load Dimension table from on-premises to Azure  
Synapse Data warehouse using SSIS

# Demo – PolyBase

1. Export table to flat file
2. Create blob storage account
3. Upload flat file to blob storage
4. Run PolyBase 6 steps process
5. Monitor and confirm successful migration
6. Confirm 60 distributions in destination table



Azure  
Synapse  
Analytics

# Azure SQL DW

## Scaling

# Azure SQL Warehouse Scaling

## Scaling in SQL Data Warehouse

- SQL DW allows us to scale or pause at will
- Scale up during heavy demand
- Pause to cut cost
- DWUs are CPU, memory, and I/O bundled into units of compute scale
- Increasing DWU's
  - increase query performance
  - Also increase maximum number of concurrent queries and concurrent slots
- Modify with GUI, PowerShell, or TSQL



# Azure SQL DW

## Backup and Restore

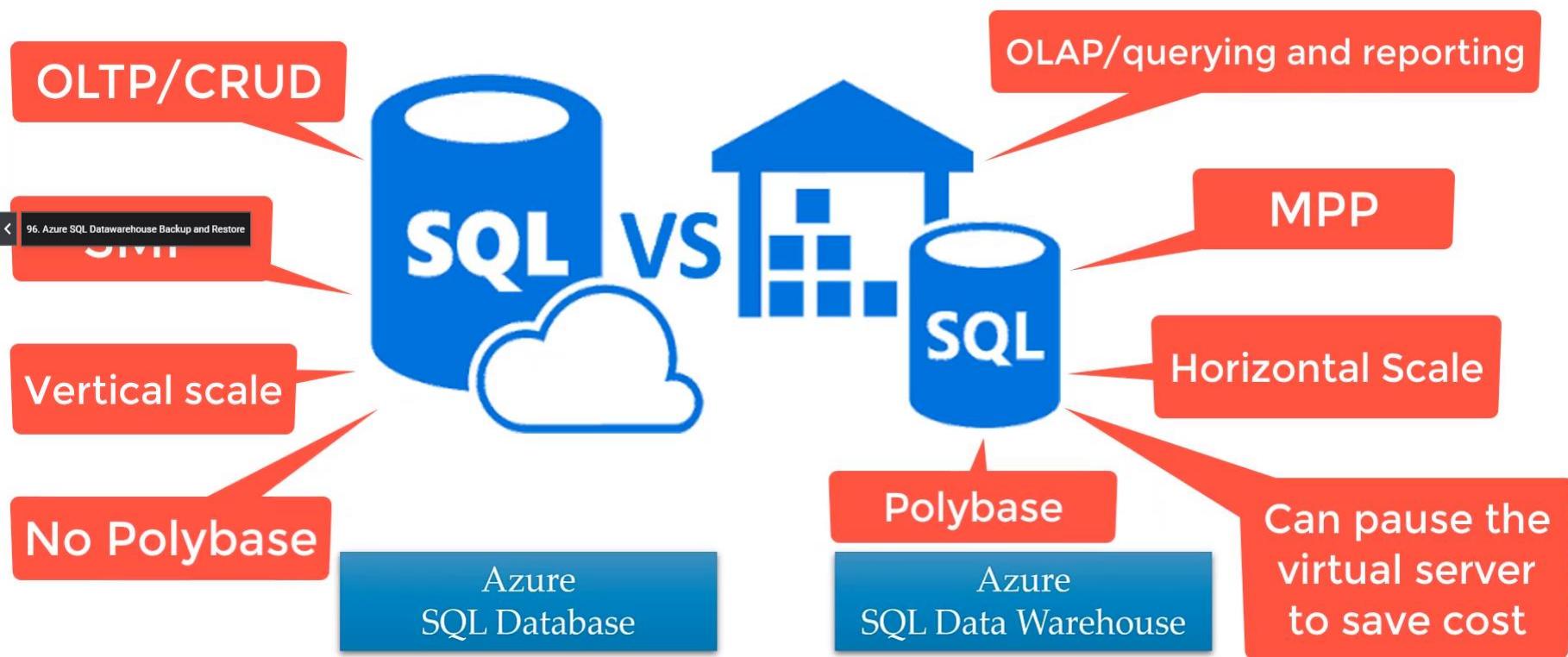
Protect your data from corruption or deletion

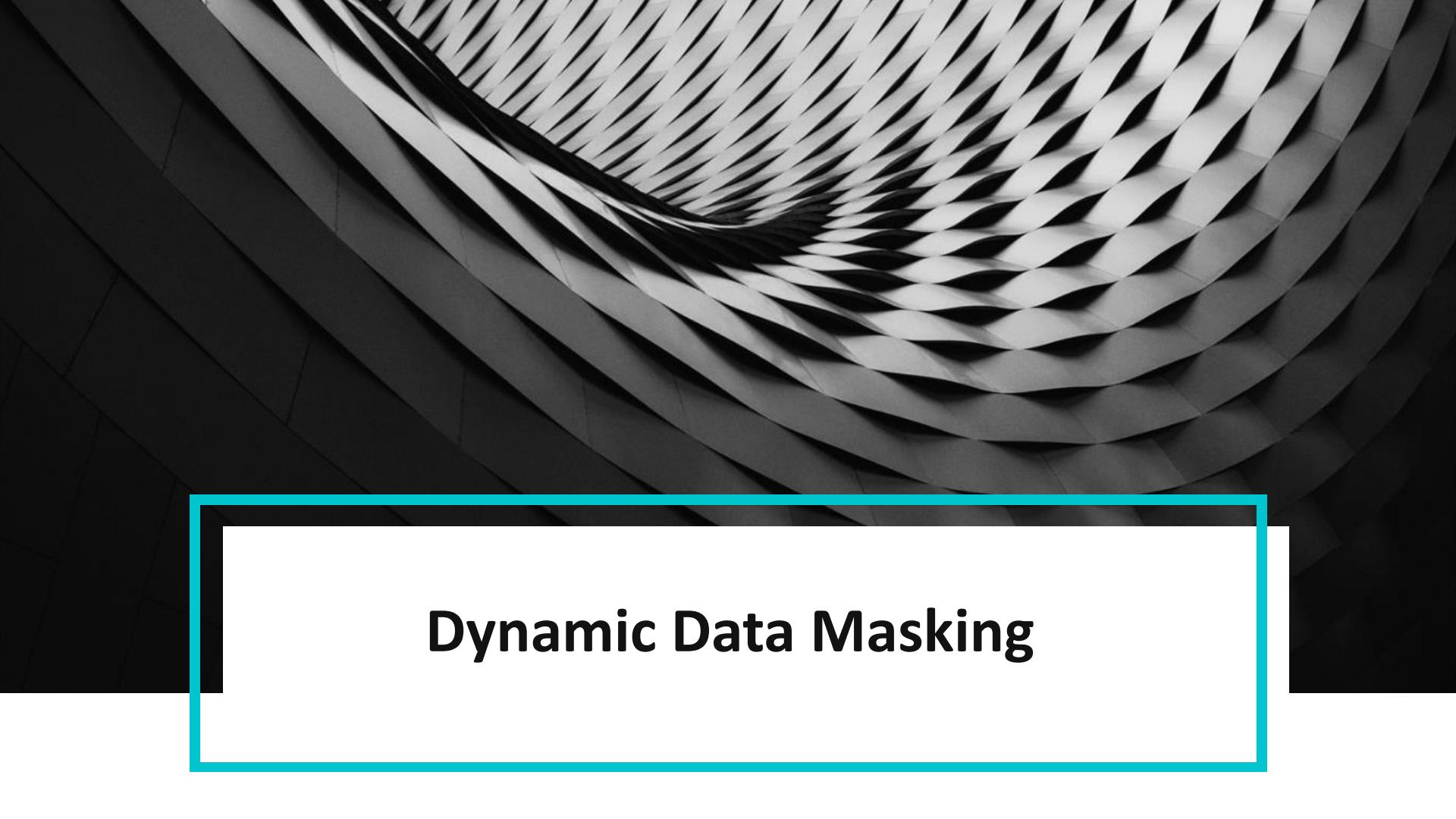
# Azure SQL DW Backup and Restore

- Snapshots of your data warehouse are taken throughout the day creating restore points
- These restore points are available for 7 days
  - Retention period of 7 days cannot be changed
- SQL pool supports an 8 hour recovery point objective (RPO)
- Replicated to paired region once a day
- You can also take user-defined snapshots
  - Retention period 7 days cannot be changed
  - 42 max restore point possible
  - Can be created using PowerShell or portal
- When you drop a SQL pool, a final snapshot is created and saved for seven days
  - SQL Pool should not be in paused state



# Azure SQL DB vs. Azure SQL DW





# Dynamic Data Masking

# Dynamic Data Masking

ID	PersonName	EmailAddress	CreditCardNumber	SocialSecurityNumber
1	Anoop Kumar	abcdefg@hotmail.com	1234-5678-4321-8765	123-45-6789
1	Rahul Gupta	amitguptaabcfgh@hotmail.com	8765-1234-5678-4321	231-45-6787
1	Amit Goel	amitgoelabcdefg@hotmail.com	4321-1234-5678-4321	321-45-6700

ID	PersonName	EmailAddress	CreditCardNumber	SocialSecurityNumber
9590	AXXXr	aXXX@XXXX.com	xxxx-xxxx-xxxx-8765	xxxx
7604	RXXXa	aXXX@XXXX.com	xxxx-xxxx-xxxx-4321	xxxx
8453	AXXXI	aXXX@XXXX.com	xxxx-xxxx-xxxx-4321	xxxx

**Random Number function** – generates random number based on selected boundaries and actual data type. Can be applied only numbers, not string

**Email function** – Exposes the first letter and replace the domain with xxx.com

**Default function** – Full masking of data. For numeric – 0 For String – XXXX characters

**Custom String function** – You can define the exposed prefix, the padding string and exposed suffix

**Credit Card function** - Only the last four digits of Credit card are shown



# Dynamic Data Masking

- **Limit the exposure of sensitive data to non-privileged users**
  - You can decide the level of exposure of data
- **No change in physical layer**
  - Data in the database is not changed
  - Not the same as data encryption
- **No additional development effort needed at application level**
- **Security: Should not be used as a primary security layer**
  - Dynamic Data Masking should not be used as an isolated measure to fully secure sensitive data
  - ad-hoc query permissions can apply techniques to gain access to the actual data.
- **Other considerations**
  - Masked columns can be updated if user has permission
  - Export masked from source data results in masked data in target table

# Dynamic Data Masking

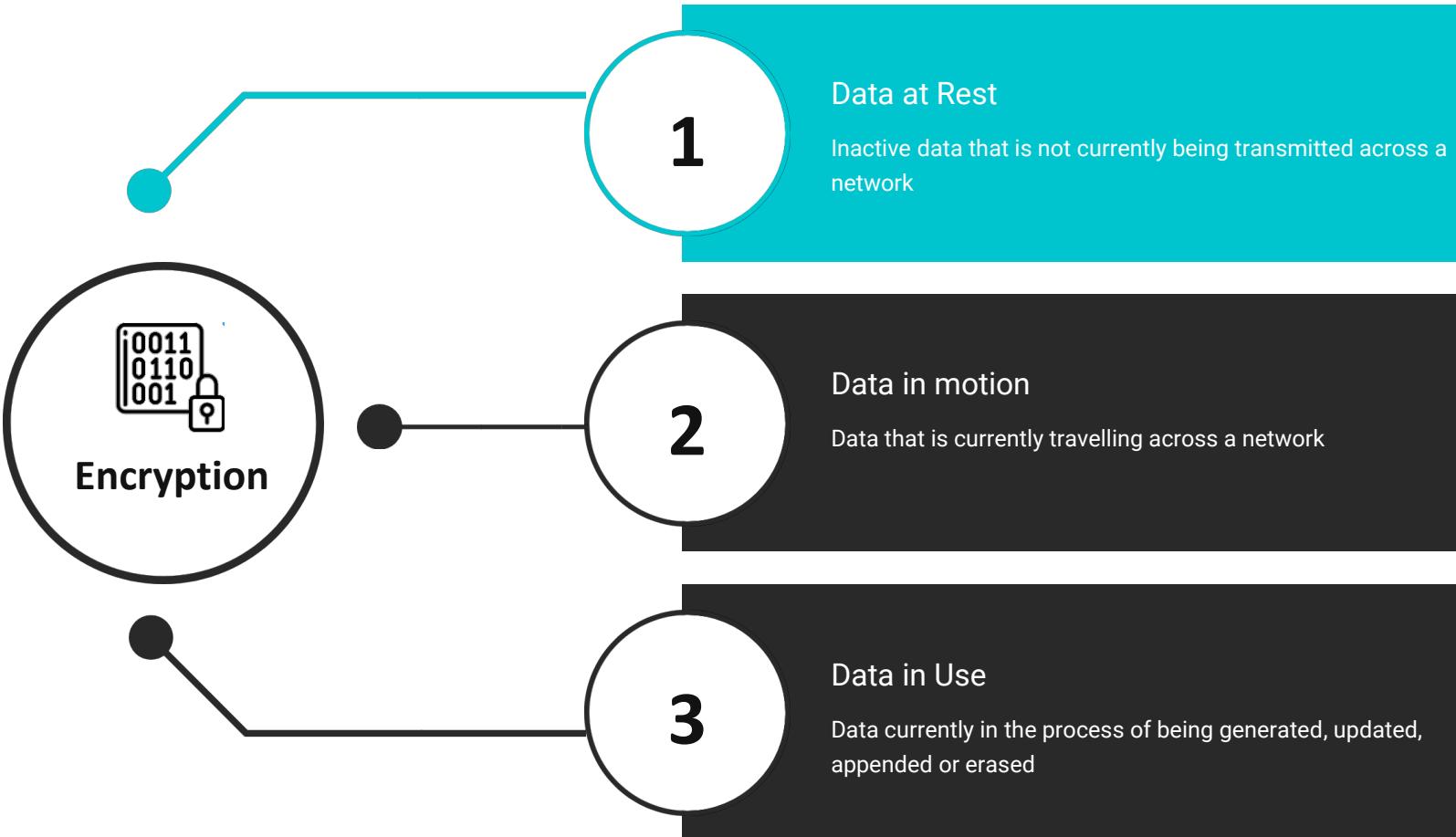


- **How to Enable DDM?**
  - Portal
  - Powershell
    - Get-AzSqlDatabaseDataMaskingRule
    - New-AzSqlDatabaseDataMaskingRule
    - Remove-AzSqlDatabaseDataMaskingRule
    - Set-AzSqlDatabaseDataMaskingRule
  - Rest API

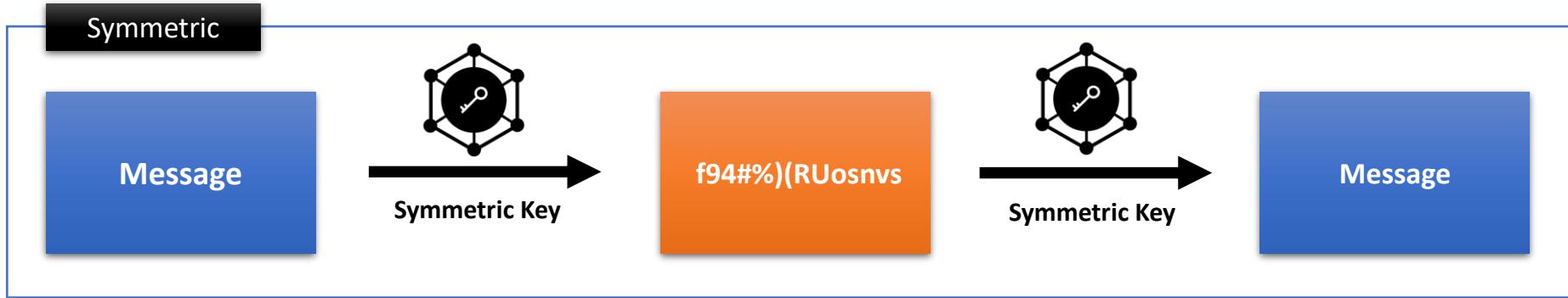


# Encryption

Of data at rest, motion or use



# Types of Encryption



Highly performant



Secure key handling is difficult

# Types of Encryption

## Asymmetric

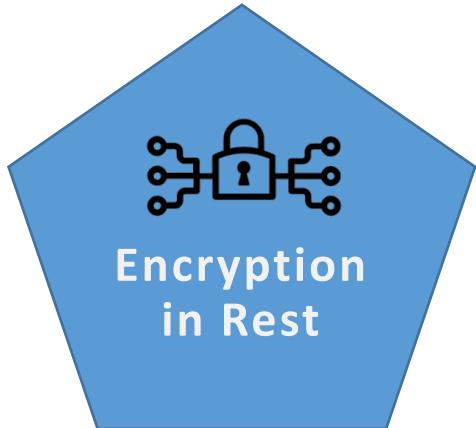


Set of keys (Public and Private) where one is used to encrypt and other to decrypt



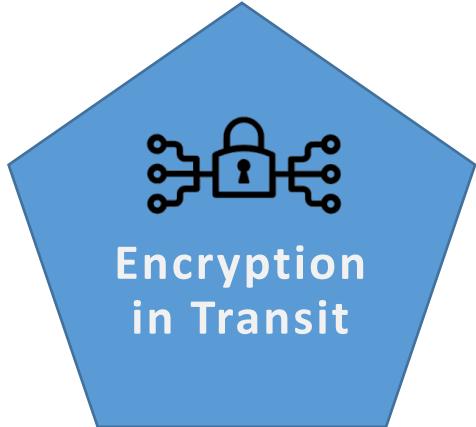
Computationally more expensive, but solves the key handling problem

# Encryption in Rest



- By default all storage services encrypted data at rest
- 256-bit AES encryption
- Keys are stored in Azure Key Vault

# Encryption in transit



- Any communication over the internet to Azure services is typically encrypted via SSL/TLS protocols
- Utilize site-to-site VPN or point-to-site VPN connections
- Or utilize ExpressRoute
- ExpressRoute communication is a private connection and not encrypted but workloads can be encrypted via SSL/TLS
- Storage services can be configured to require secure transfer

# Types of Encryption



## Deterministic encryption:

- This will always generate the same encrypted value for any plain text value.
- You can perform point lookups, equality joins, grouping and indexing on encrypted columns.

## Randomized encryption:

- This is more secure than deterministic encryption because the encrypted value is generated in a less predictable manner.
- But you can't perform searching, grouping, indexing or joins on encrypted columns.

# Types of Encryption



## Column Encryption Key (CEK)

- Used to encrypt values in specific columns
- Encrypted versions of each CEK is stored in the database.

## Column Master Key (CMK)

- Used to encrypt all the CEKs
- Must be stored externally in a secure key store
- Key store providers: Azure key vault, certificate store, HSM