Microsoft Azure

Azure SQL Database

Chapters

Azure SQL Database Overview
Evolution of SQL Server
Benefits
Improvements
Business Model
ROI

Intelligent Performance
Single SQL Code Base
In-Memory
Query Store
xEvents
Query Performance Insights
Autotuning
Monitoring at scale
Adaptive query processing

Scale on the fly
Scalable performance tiers
Predictable Performance
Real time performance analytics
Optimize resources

Business continuity
Recovery
Geo-replication
Azure Data Sync
Built-in HA
Zone Redundancy

Deployment options & tiers
Managed Instance
Hyperscale
Elastic pool
Serverless

Secure and protects
Encryption
Transparent Data Encryption
Vulnerability Assessment
Information Protection
Row Level Security
Dynamic Data Masking
Threat Detection
Auditing

Programmatic Capability
Graph
Machine Learning Services

Migrate to Azure SQL Database

SaaS Patterns and Reference Architectures

Customer evidence

Azure SQL Database

The developer's intelligent cloud database service

Learning Objectives

Azure SQL Database overview

Evolution of Azure SQL Database

Key benefits

Key improvements

Business model

Azure SQL Database ROI

The developer's intelligent cloud database service

Built for application developers

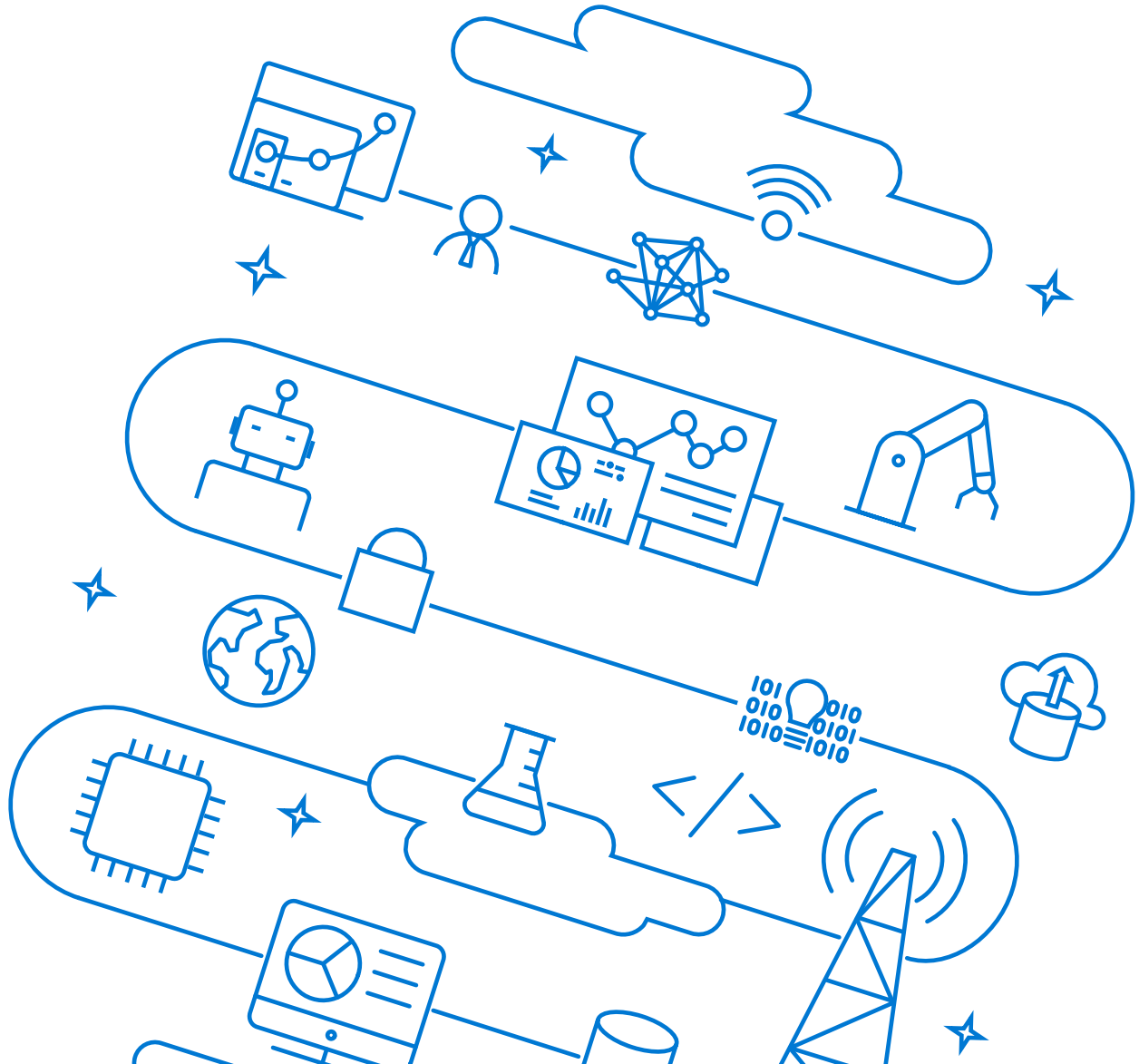
Gives developers more time to innovate

Accelerates time to market

Maximizes performance, reliability, and data protection

Allows developers to use the languages and platforms they prefer

Everything built-in



The Modern Data Problem

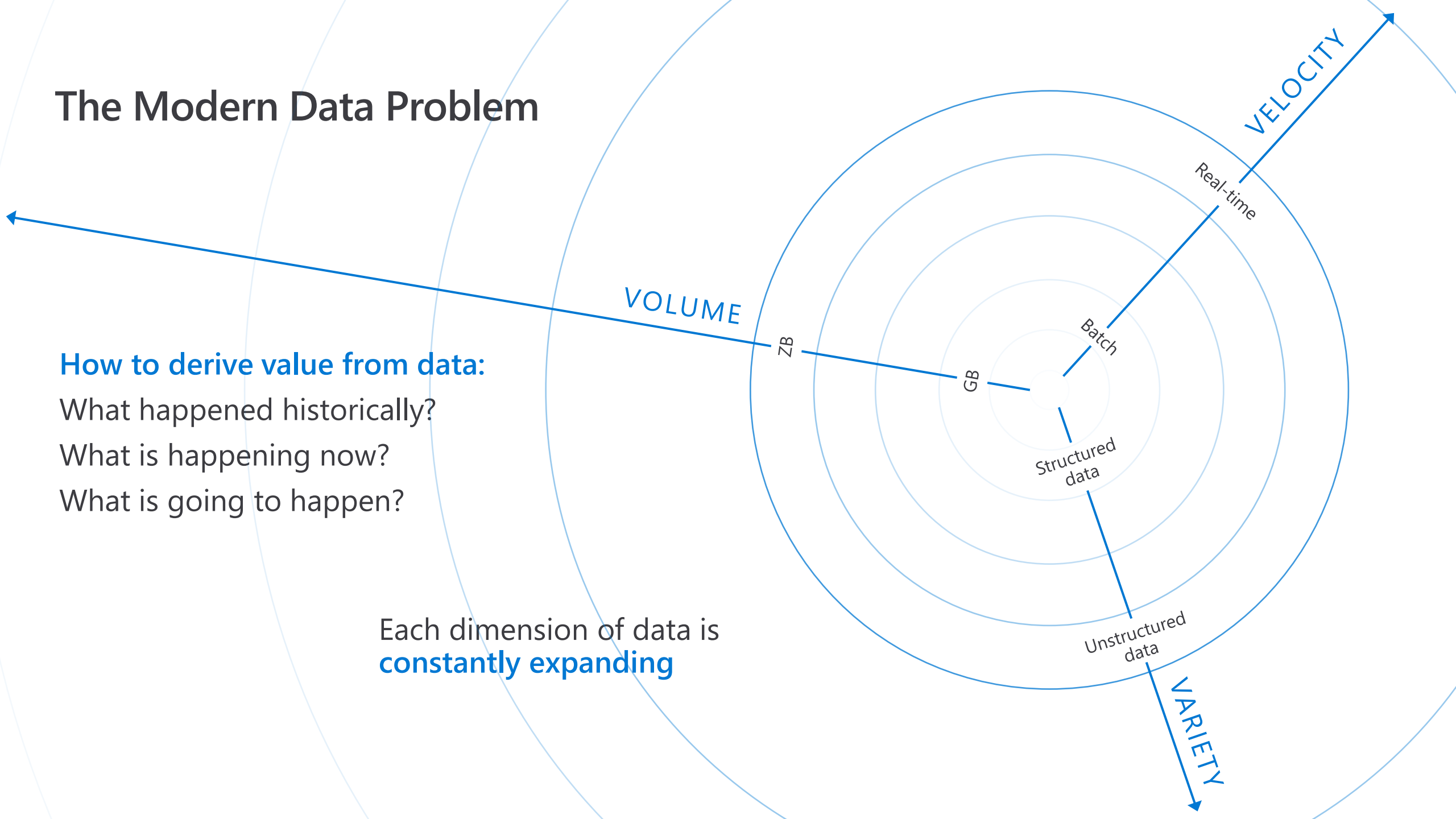
How to derive value from data:

What happened historically?

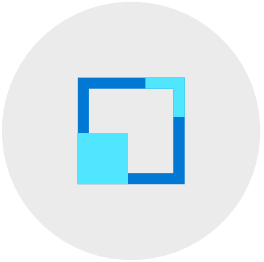
What is happening now?

What is going to happen?

Each dimension of data is **constantly expanding**



Manageability with Azure SQL Database



Elastic scale and performance

- Predictable performance levels
- Programmatic scale out
- Dashboard views of database metrics

Business continuity & data protection

- Seamless hybrid deployments
- Disaster recovery
- Compliance enabled
- Always On High Availability
- Advanced Threat Protection

Familiar and fully-managed

- Easy-to-use tools
- Flexible languages
- Near-zero administration

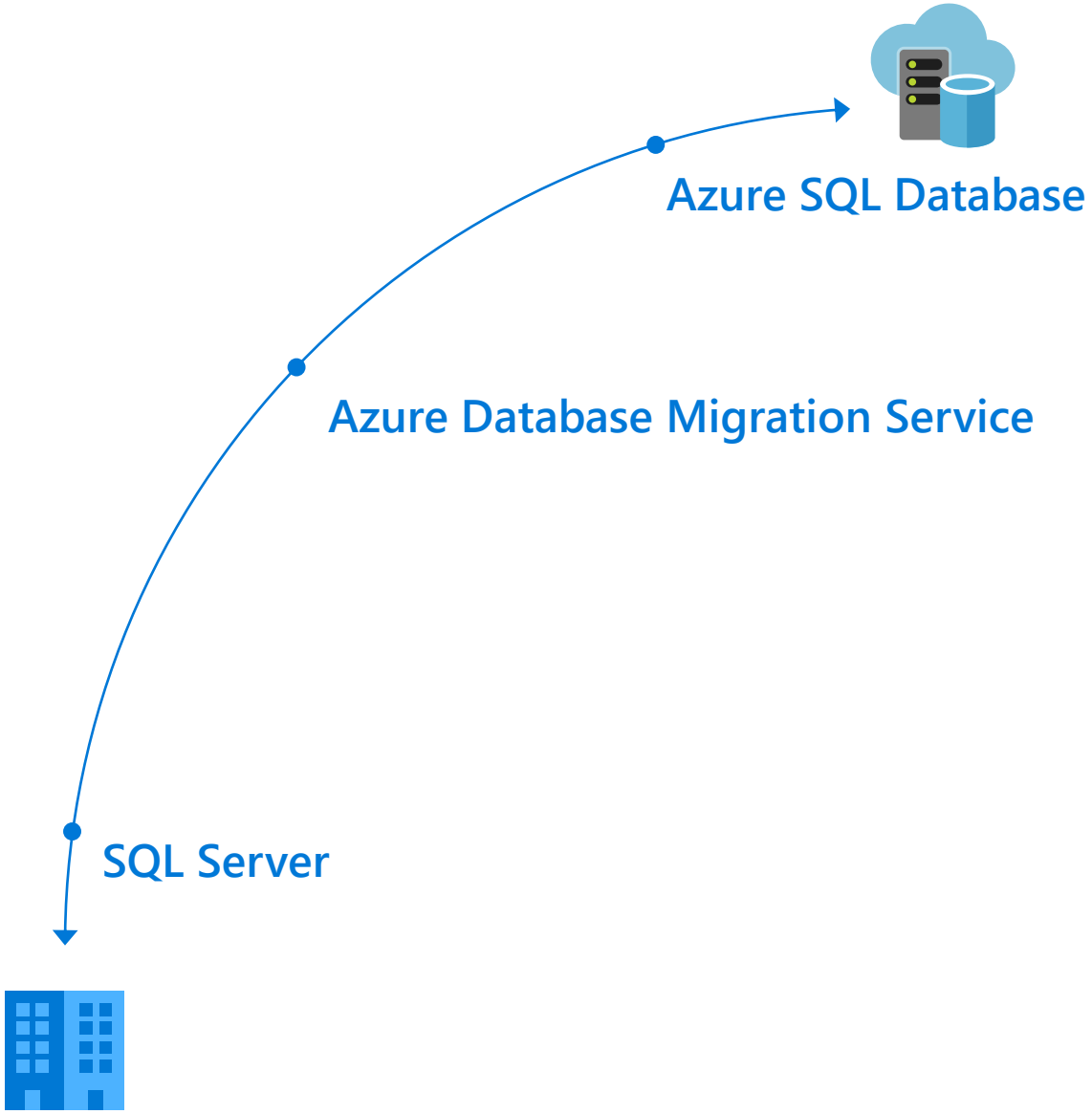
A hybrid Journey to the Cloud

Seamless hybrid deployment with integrated data synchronization

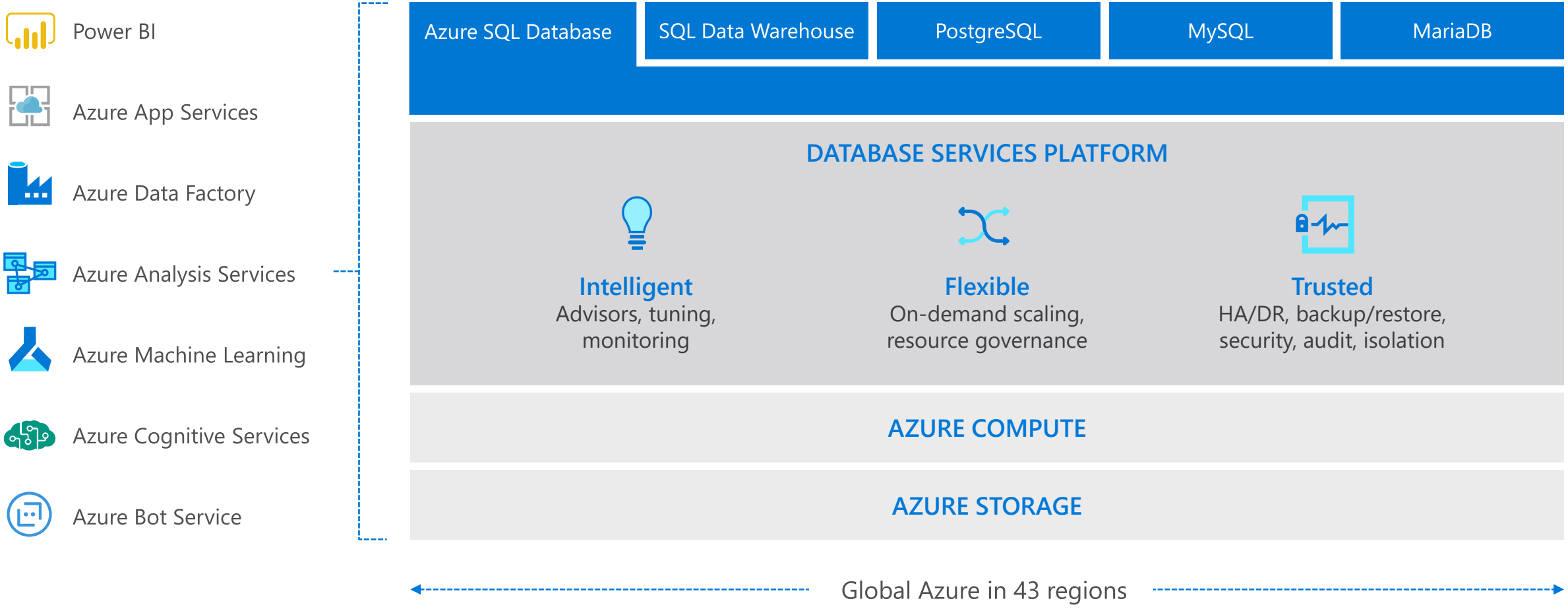
Reliable migration at scale

Lift and shift to the cloud with no code changes

Up to 55% cost savings



Azure Relational Database Platform



Azure SQL Database deployment option

Azure SQL Database

Single

Database-scoped deployment option with predictable workload performance

Best for apps that require resource guarantee at database level

Elastic Pool

Shared resource model optimized for greater efficiency of multi-tenant applications

Best for SaaS apps with multiple databases that can share resources at database level, achieving better cost efficiency

Managed Instance

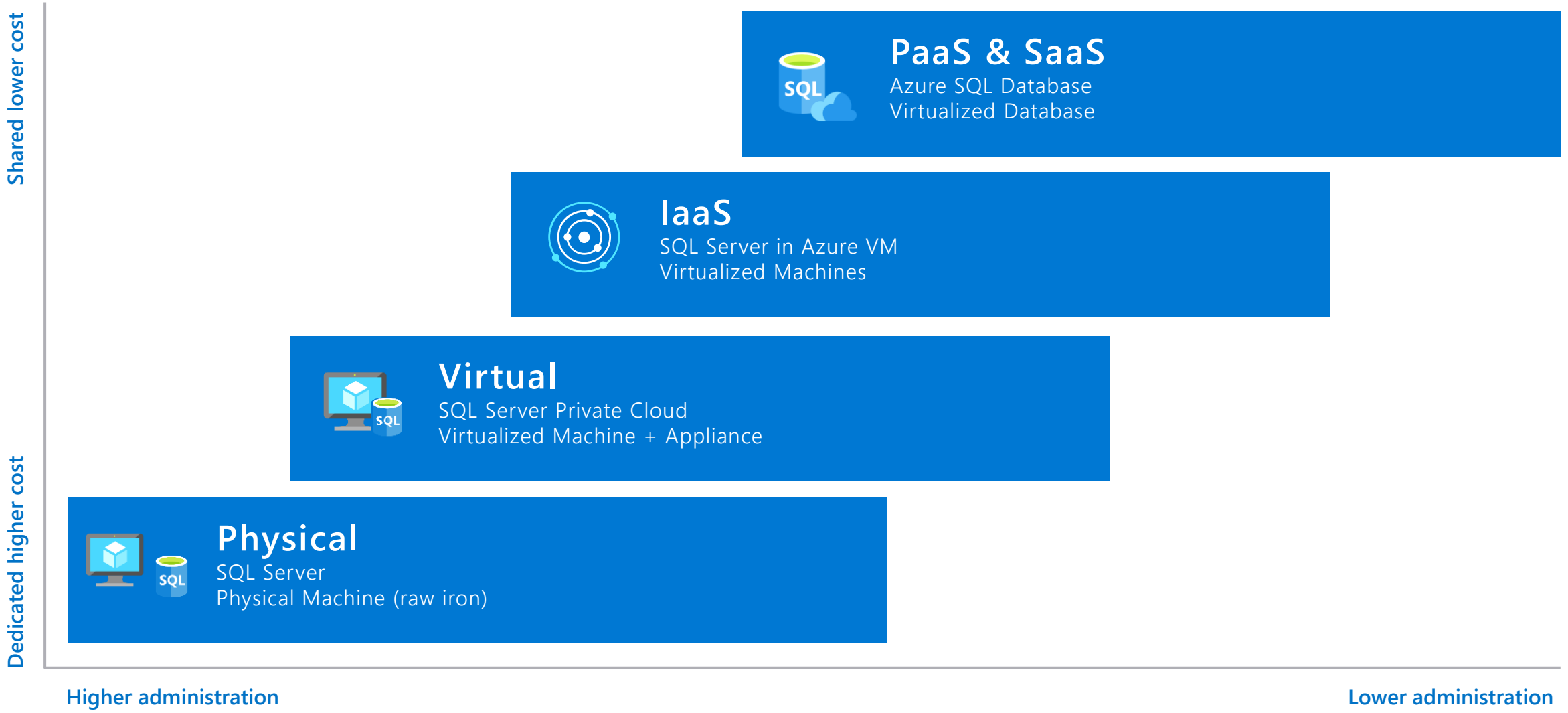
Instance-scoped deployment option with high compatibility with SQL Server and full PaaS benefits

Best for modernization at scale with low friction and effort

Service Tiers

- General Purpose
- Business Critical
- Hyperscale
- Serverless

Data platform continuum



Key benefits of Azure SQL Database

Independently scale compute and storage to match both performance and financial needs

High availability and disaster recovery with 99.99% uptime availability SLA and active-geo replication, point-in-time restore, and geo-restore

Up to 100x performance improvements with support for In-Memory Columnstore queries

Improved monitoring and troubleshooting with Extended Events and visibility into more than 100 new table views

Support for key programmability functions to drive more robust application design

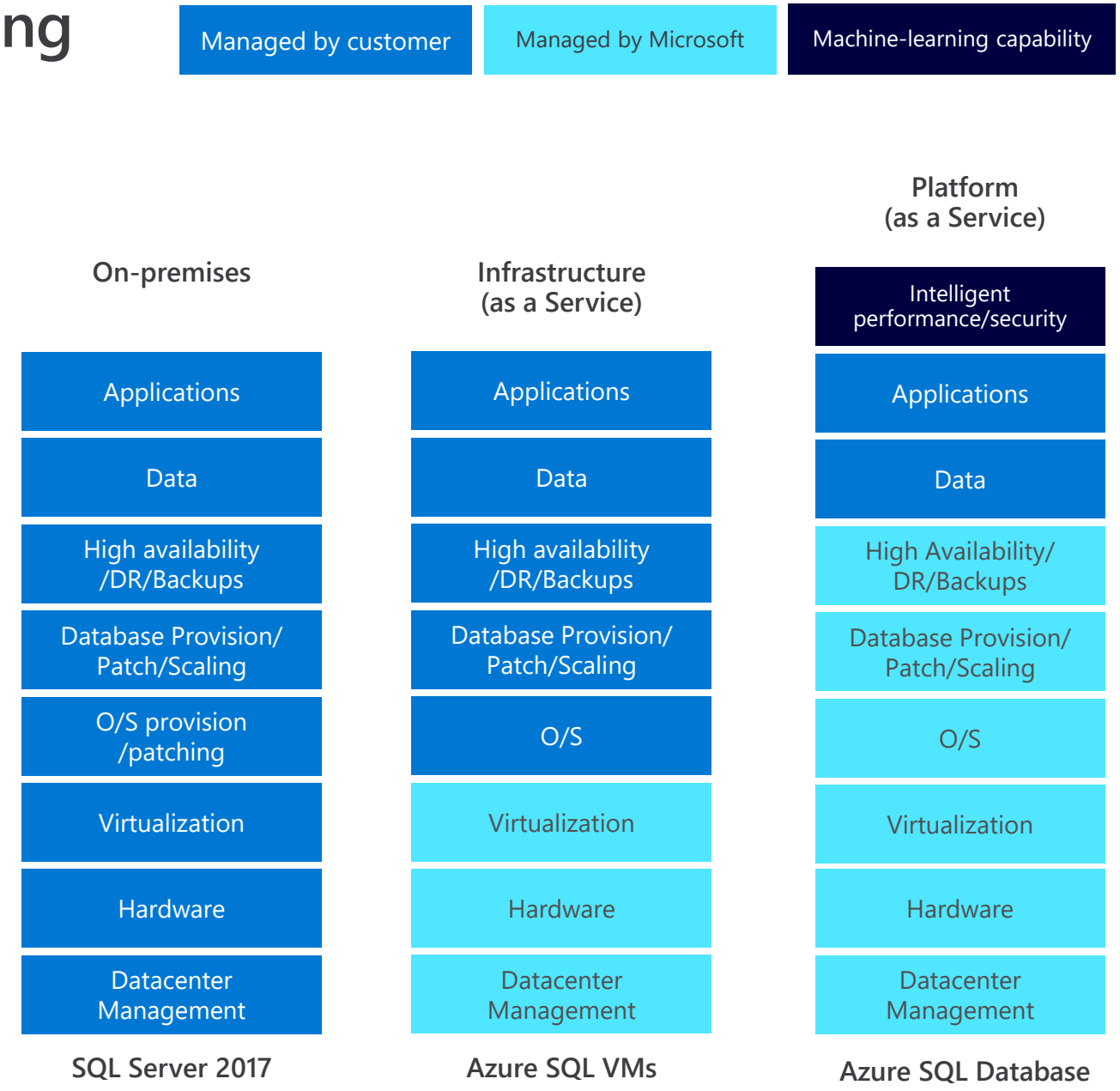
Saving opportunity for modernizing your data estate is significant

On-premises costs tend to be driven by hardware and data center management costs

Infrastructure-as-a-Service reduces cost categories related to data center and compute

Platform-as-a-Service off-loads customers' most administrative tasks to Azure, further improving efficiency with machine-learning capabilities for performance and security

- **Managed Instance**: instance-level deployment for lift-shift existing apps to Azure, fully backward compatible
- **Single database**: database-level deployment for new apps



Focus on your business

Your work so far	How PaaS helps
Hardware purchasing and management	**Built-in** scale on-demand
Protect data with backups (with health checks and retention)	**Built-in** point-in-time restore
High availability implementation	**Built-in** 99.99% SLA and auto-failover
Disaster recovery implementation	**Built-in** geo-redundancy and geo-replication
Ensure compliance with standards on your own	**Built-in** easy to use features
Secure your data from malicious users and mistakes	**Built-in** easy to use features
Role out updates and upgrades	**Built-in** updates and upgrades
Monitor, troubleshoot, and manage at scale	**Built-in** easy to use features
Tune and maintain for predictable performance	**Built-in** easy to use features

We take care of your database chores

Updates to Azure SQL Database

Hybrid

Azure Hybrid Benefit
Managed Instance Business Critical – GA

Managed Instance General Purpose - GA

vCore Purchasing Model - GA

Reserved Capacity Pricing - GA

Data Sync – GA

More vCore compute levels - GA

Performance & scale

Zone Redundancy - Prev

Elastic DB library for Java - GA

Columnstore in Standard tier - GA

Long Term backup retention - GA

Zone Redundancy - GA

Read Scale - Prev

.Net/ODBC and SSDT - Prev

Resumable Online Index Create – GA

Storage add-ons - GA

Adding DTU Standard perf levels- GA

Elastic Jobs - Prev

Auto Failover - GA

Hyperscale – Prev

Dev/test pricing for MI, single DB- GA

Serverless - Prev

Security

TDE with Azure Key Vault – GA

Information Protection – Prev

Vulnerability Assessment - GA

Intelligence

Automatic Tuning Improvements - GA

Intelligent QP updates - Prev

For latest information:
https://azure.microsoft.com/en-us/updates/?product=sql-database
https://docs.microsoft.com/en-us/azure/sql-database/sql-database-release-notes

Previous updates to Azure SQL Database

Learn & Adapt

Operational analytics
- Columnstore
- In-Memory OLTP

Predictable performance
- Query Store
- Index Optimization
- Automatic tuning
- Auto query plan correction
- Performance Insight in OMS
- Adaptive Query Processing

SQL Graph Advanced analytics
- Native PREDICT
- R Services

Privacy & trust

Activity monitoring
- Engine Audit
- Threat Detection
- Centralized dashboard OMS

Access control
- SQL Firewall
- RLS, Dynamic data masking
- AAD and MFA

Data protection
- Encrypt in motion (TLS)
- Always Encrypted (equality)
- TDE & BYOK
- Service endpoint
- Always Encrypted (secure enclave)

Discovery & assessment
- Vulnerability assessment

Business Continuity

HA-DR built-in
- 99.99% SLA
- Geo-restore
- Active geo replicas (4)
- Multi-AZ
- Zone-redundant

Backup and restore
- Backup with health check
- 35 days PITR
- 10 years data retention

Distributed application
- Change Tracking
- Transaction replication
- Data sync
- SSIS service
- Read scale-out
- VNET endpoints

Seamless and Compatible

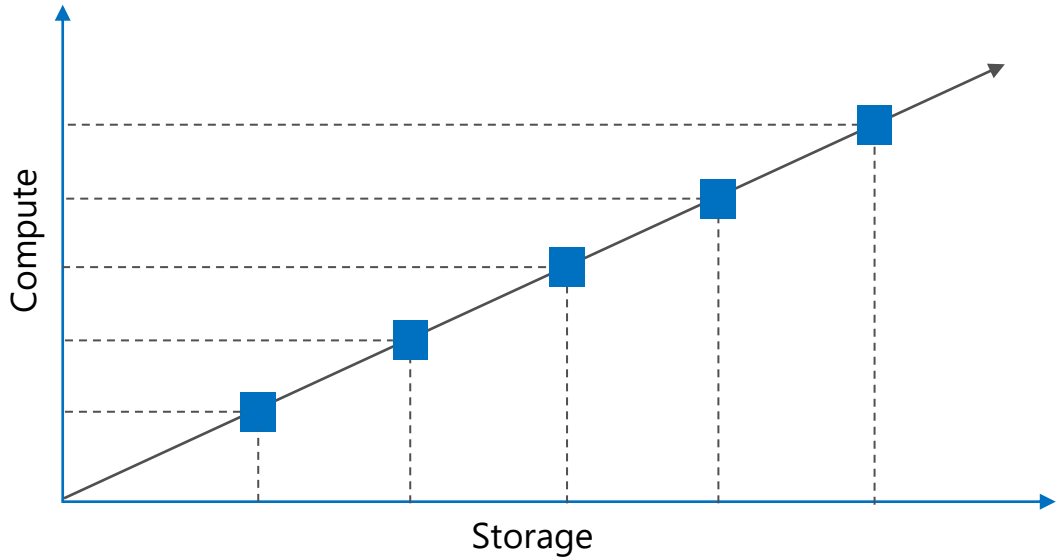
Biz model & SKUs
- DTU/eDTU
- <=1TB
- Bigger std: S4-S12
- Separate compute and storage
- Azure Hybrid Benefit
- vCore-based purchasing

Cost optimization
- Intelligent PaaS

Flexible compute & storage options

DTU model

Simple, preconfigured



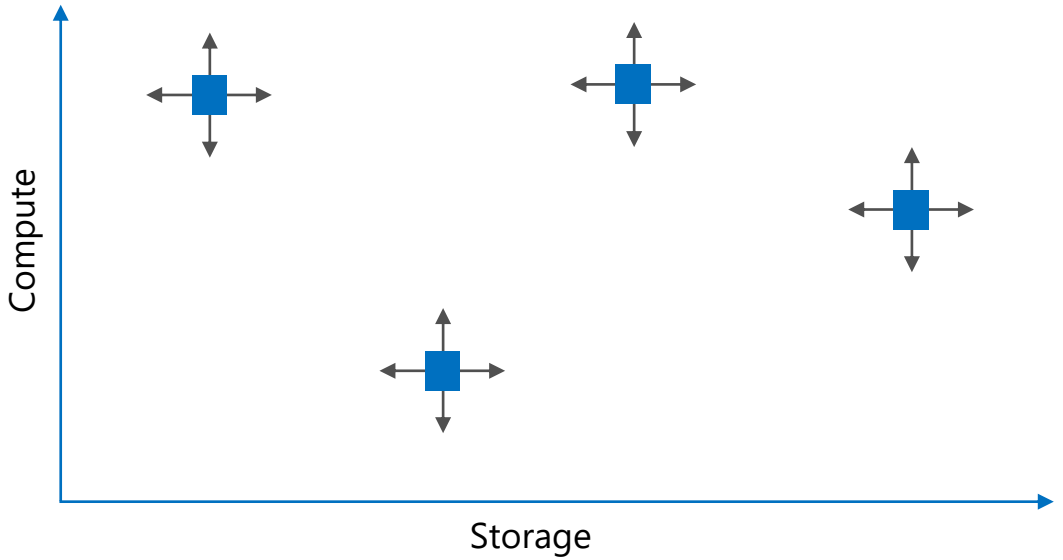
Pre-packaged, bundled unit that represents the database power

Designed for predictable performance, but somewhat inflexible and limited in options

DTU sizing offers simplicity of choice

vCore model

Independent scalability



This model allows you to independently choose compute and storage resources. It also allows you to use Azure Hybrid Benefit for SQL Server to gain cost savings.

Best for customers who value flexibility; control and transparency

Benefits of virtual cores

Choice between vCores and DTUs in Azure SQL Database as a unit of measure for available CPU

Understand your compute requirements in the cloud vs. what you use on-premises today

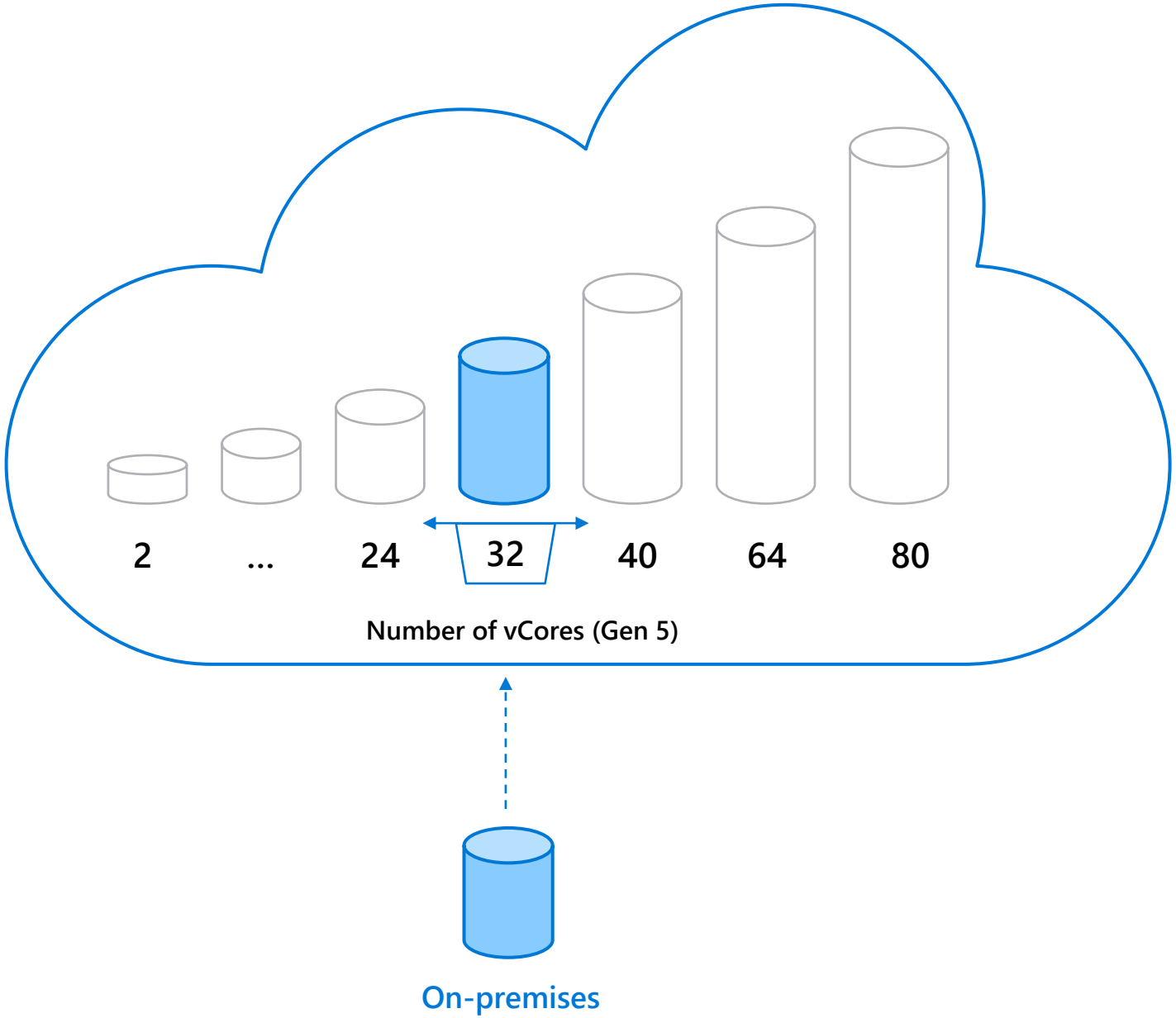
Easier to right-size the destination environment by removing the guesswork of DTUs

Gen 4 CPUs are based on Intel E5-2673 v3 (Haswell) 2.4 GHz processors. In Gen 4, 1 vCore = 1 physical CPU

1-24 vCores

Gen 5 logical CPUs are based on Intel E5-2673 v4 (Broadwell) 2.3 GHz processors. In Gen 5, 1 vCore = 1 hyper thread

2-80 vCores



Azure Hybrid Benefit for SQL Server

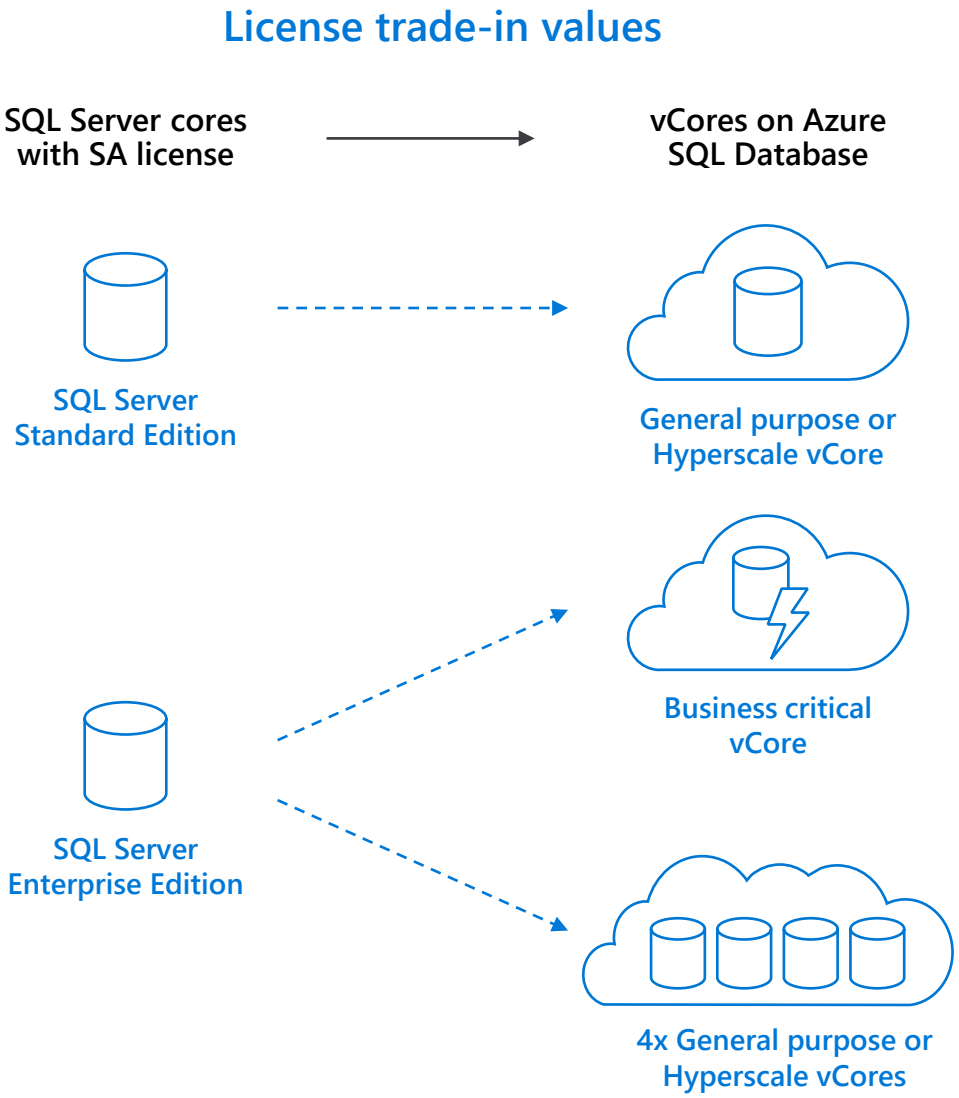
Take an inventory of on-premises licenses to determine potential for conversion

Convert on-premises cores to vCores to maximize value of investments

1 Standard license core =
1 General Purpose or Hyperscale core

1 Enterprise license core =
1 Business Critical core

1 Enterprise license core =
4 General Purpose or Hyperscale cores
(virtualization benefit)



Flexible compute, storage & performance options

Simplicity

We remain committed to the DTU-based model and the simplicity it offers customers who want a pre-configured solution

Flexibility:

The vCore-based model reflects our commitment to customer choice and to simplify the hybrid benefit for customers migrating from on-premises

Customers pay for:

Service tier + number of vCores

Type and amount of data storage

Number of IO

Backup storage (RA-GRS)

Service tier	General purpose		Business critical		Hyperscale
Best for	Most **budget-oriented** workloads		Critical business applications with **high IO requirements**.		**VLDB OLTP and HTAP workloads** with highly scalable storage and read-scale requirements
Deployment option	Single / Elastic Pools	Managed Instance	Single / Elastic Pools	Managed Instance	Single
Compute tiers	Gen4: 1 to 24 vCore Gen5: 2 to 80 vCore	Gen4: 4 to 24 vCore Gen5: 4 to 80 vCore	Gen4: 1 to 24 vCore Gen5: 2 to 80 vCore	Gen4: 4 to 24 vCore Gen5: 4 to 80 vCore	Gen4: 1 to 24 vCore Gen5: 2 to 80 vCore
Storage	**Premium remote**		**Local SSD**		**Local SSD Cache**
	32GB – 8TB per instance	32GB – 8TB per instance	32GB – 4TB per instance	32GB – 8TB per instance	Scale from 5GB to 100TB of storage in 1GB increments
In-Memory	Not supported		Supported		Not supported
Read-write IO	~2ms for all data access		<0.5ms for all data access		<0.5ms for hot data access ~2ms otherwise
Availability	2 read replicas		3 replicas, 1 read-scale replica, zone-redundant HA		Primary read/write replica + up to 4 read replicas
Backups	RA-GRS, 7-35 days (7 days by default)		RA-GRS, 7-35 days (7 days by default)		LRS, ZRS, RA-GRS, 7-35 days (7 days by default)

For latest information reference: https://azure.microsoft.com/en-us/pricing/details/sql-database/

Choose from two hardware generations

Balance performance requirements and price with two hardware generations

Match your on-premise application behavior

	Gen 4	Gen 5
Hardware	Intel E5-2673 v3 (Haswell) 2.4 GHz processors vCore = 1 PP (physical core)	Intel E5-2673 v4 (Broadwell) 2.3 GHz processors, fast eNVM SSD vCore=1 LP (hyper-thread)
Performance levels	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 16, 24 vCores	2, 4, 6, 10, 12, 14, 16, 18, 20, 24, 32, 40, 80 vCores
Memory	7 GB per vCore	5.1 GB per vCore
Storage	5 GB to 4 TB with 1 GB increments. Premium blob storage	5 GB to 4 TB with 1GB increments. Local SSD storage. Hyperscale tier 5 GB to 100 TB with 1GB increments and only charge for storage based on usage.

For latest information reference: https://azure.microsoft.com/en-us/pricing/details/sql-database/

Pay only for what you need

DTUs			vCores		
Basic	**Standard**	**Premium**	**General Purpose**	**Business Critical**	**Hyperscale**
Small databases particularly those in development phases	General purpose databases with moderate performance requirements	Mission-critical databases with high performance and high-availability requirements	Data applications with basic IO and basic availability requirements	Business critical data applications with fast IO and high availability requirements	VLDB OLTP and HTAP workloads with highly scalable storage and read-scale requirements

Elastic scale and performance: Three service tiers within DTU-based model, and two tiers within vCore-based model let you scale up and down based on throughput needs, and offer better resource isolation and an improved billing experience

Business continuity and data protection: A spectrum of business-continuity features across tiers lets you dial up control over data recovery and failover

Familiar and fully-managed: Near-complete SQL Server compatibility and unprecedented efficiencies as your applications scale with a near-zero maintenance service and a variety of familiar management tools and programmatic APIs

Azure SQL Database ROI

ROI summary for Azure SQL Database as a service (DaaS)

406%

Five-year ROI

$18,784

Five-year total average business benefit per Azure SQL Database

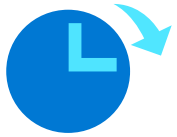
8.6 mo

Payback period

Source: IDC White Paper | The Business Value of Microsoft Azure SQL Database Services
http://download.microsoft.com/download/A/1/D/A1D5C24B-284A-4F9C-B121-E6E5FDAB1B94/IDC-Azure-SQL-DB-Value-2015.pdf

Reduce unplanned downtime with Azure SQL Database

Improvement percentages using Azure SQL Database



71%

Reduction in instances of unplanned downtime per year

91.1%

Reduction in mean time to repair (MTTR) for unplanned downtime (hours)

98%

Reduction in productive time lost (hours)

Source: IDC White Paper | The Business Value of Microsoft Azure SQL Database Services
(http://download.microsoft.com/download/A/1/D/A1D5C24B-284A-4F9C-B121-E6E5FDAB1B94/IDC-Azure-SQL-DB-Value-2015.pdf)

Save money with Azure SQL Database ROI

Average annual benefits per Azure SQL Database

$2,704

IT staff productivity benefits

$2,428

IT infrastructure cost reductions

$164

Risk mitigation/ user productivity benefits

$159

Business productivity benefits

Source: IDC White Paper | The Business Value of Microsoft Azure SQL Database Services (http://download.microsoft.com/download/A/1/D/A1D5C24B-284A-4F9C-B121-E6E5FDAB1B94/IDC-Azure-SQL-DB-Value-2015.pdf)

Azure SQL Database — Everything built-in

Intelligent performance

Realize automatic performance improvements from continuous assessment and innovation

Scales on the fly

Change service tiers, performance levels, and storage dynamically without downtime

Business continuity

Easily manage and monitor business critical functions for reliable operations

Works in your environment

Develop your app and connect to SQL Database with the tools and platforms you prefer

Advanced threat pprotection

Build security-enhanced apps with built-in protection and industry-leading compliance

The Total Economic Impact of Azure SQL Database Managed Instance

Microsoft commissioned Forrester Consulting to conduct a Total Economic Impact™ study to examine potential cost savings and business benefits enterprises would achieve from migrating on-premises workloads to Azure SQL Database Managed Instance.

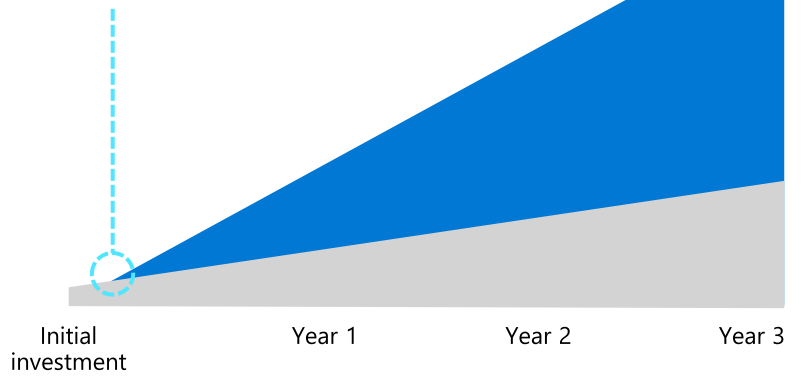
Key report benefits and findings

212% Total ROI

3 years after initial investment & migration to Azure SQL Database Managed Instance

6 months or less

Payback post migration



$1.8M

Savings from avoided hardware, network, storage and ongoing maintenance costs

40%

Improvement in productivity for in-house DBAs

20%

Increase in productivity for IT teams

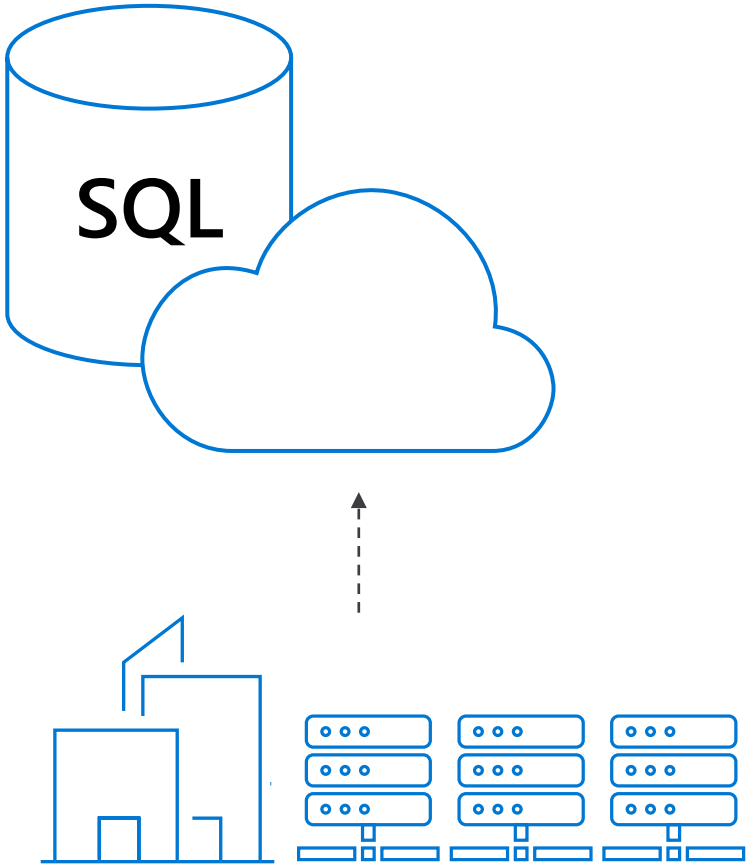
> **Azure SQL Database Managed Instance** is part of our strategic mandate to move all of our application, database, and services footprint to the Azure cloud. **We can quickly integrate and are more nimble and more efficient as a result.**
>
> *Head of development, technology company*

Download the full Total Economic Impact™ of Azure SQL Database Managed Instance report

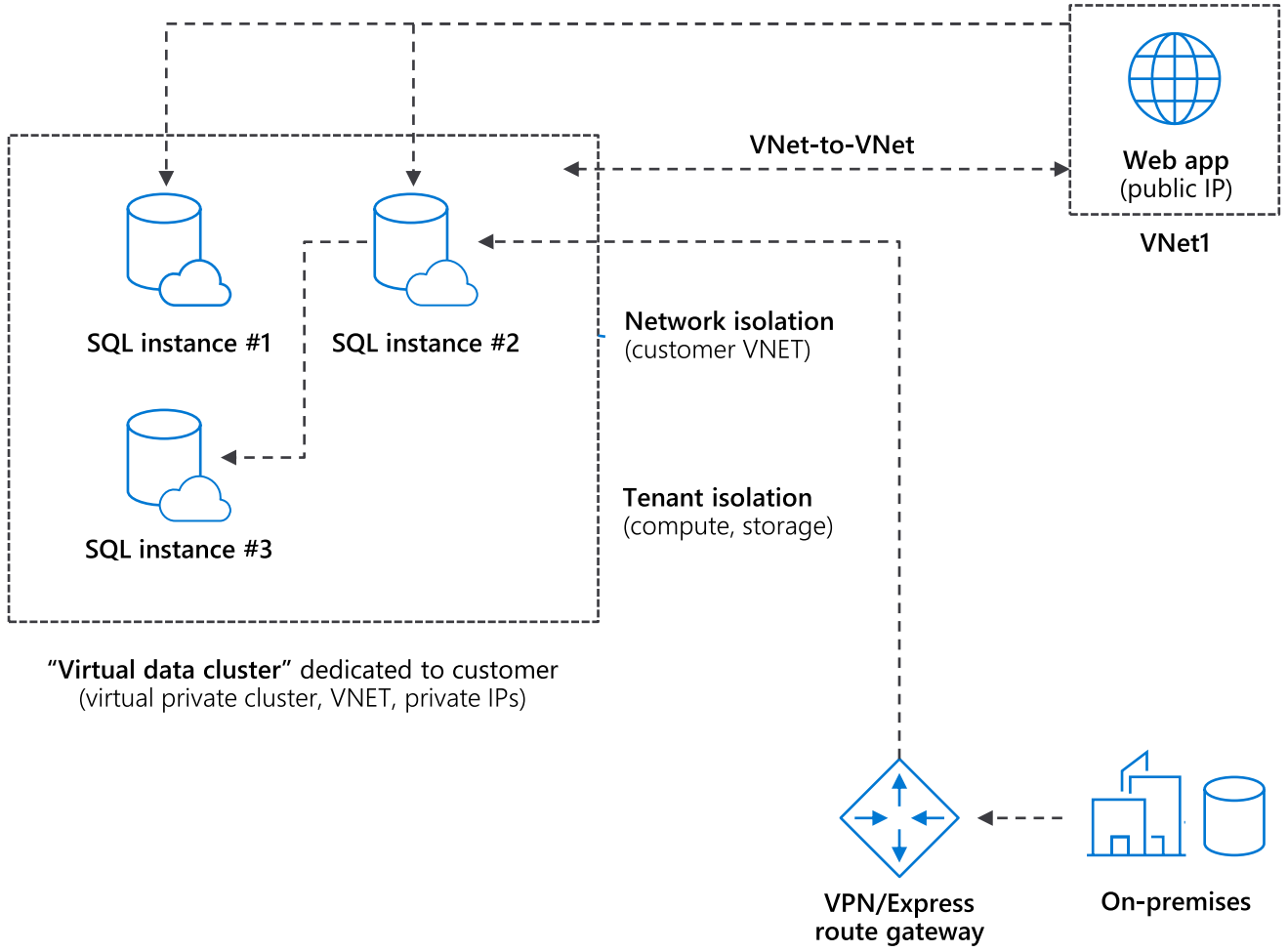
Azure SQL Database Managed Instance

Who is Managed Instance for?

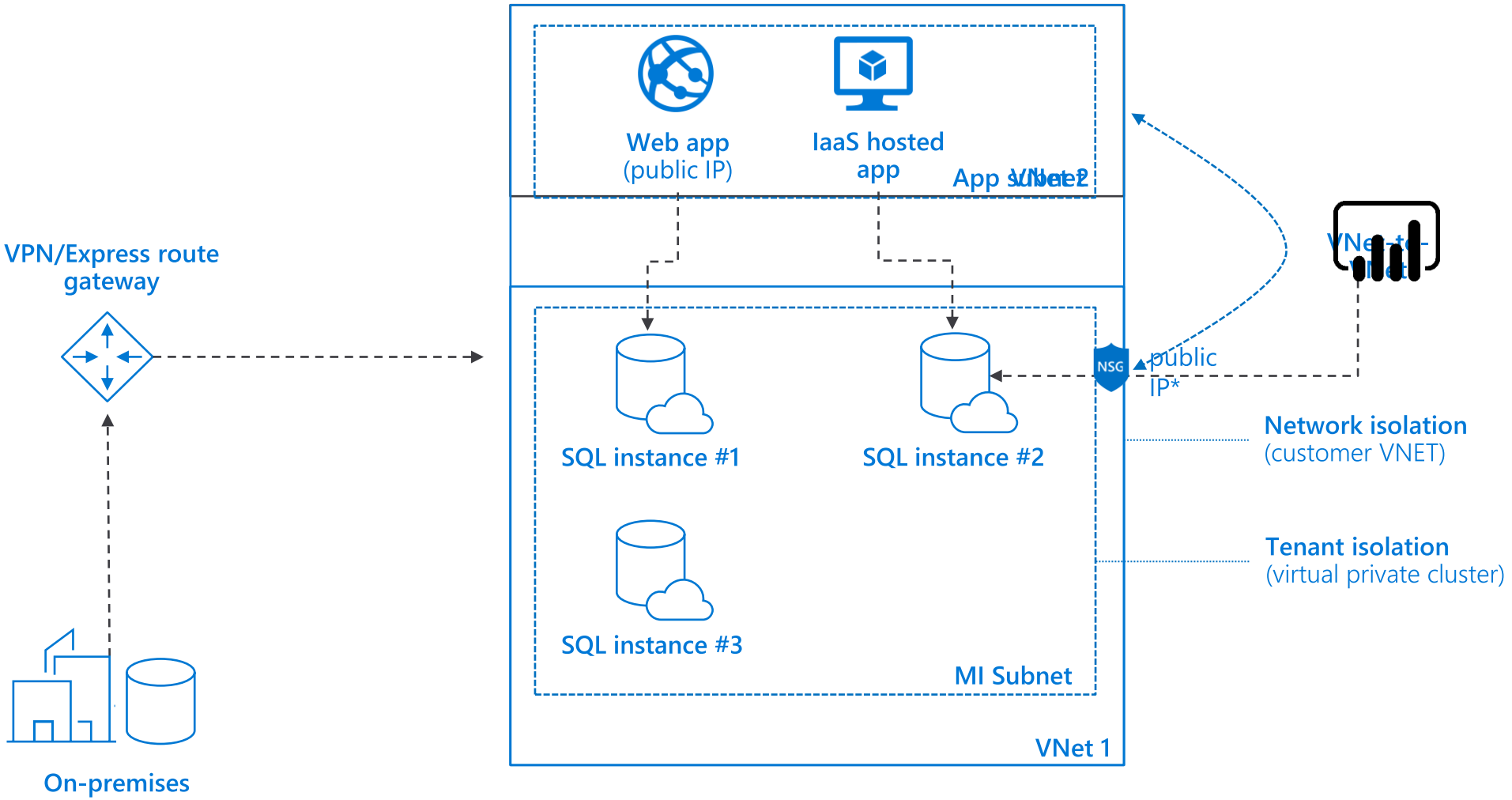
Customers looking to migrate a large number of apps from on-premise or IaaS, self-built or ISV provided, with as low migration effort as possible & cost being a crucial factor



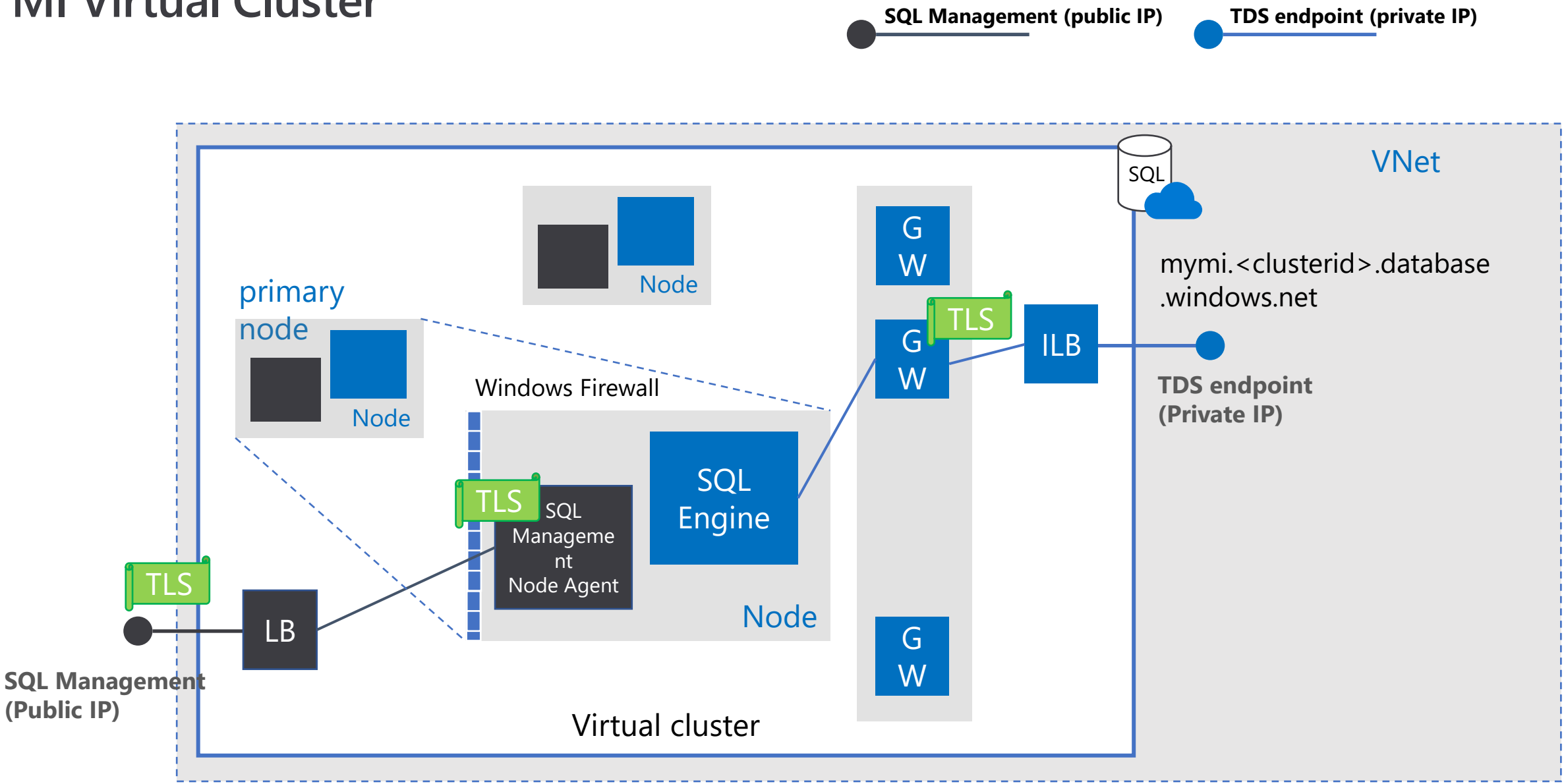
Dedicated resources through customer isolation



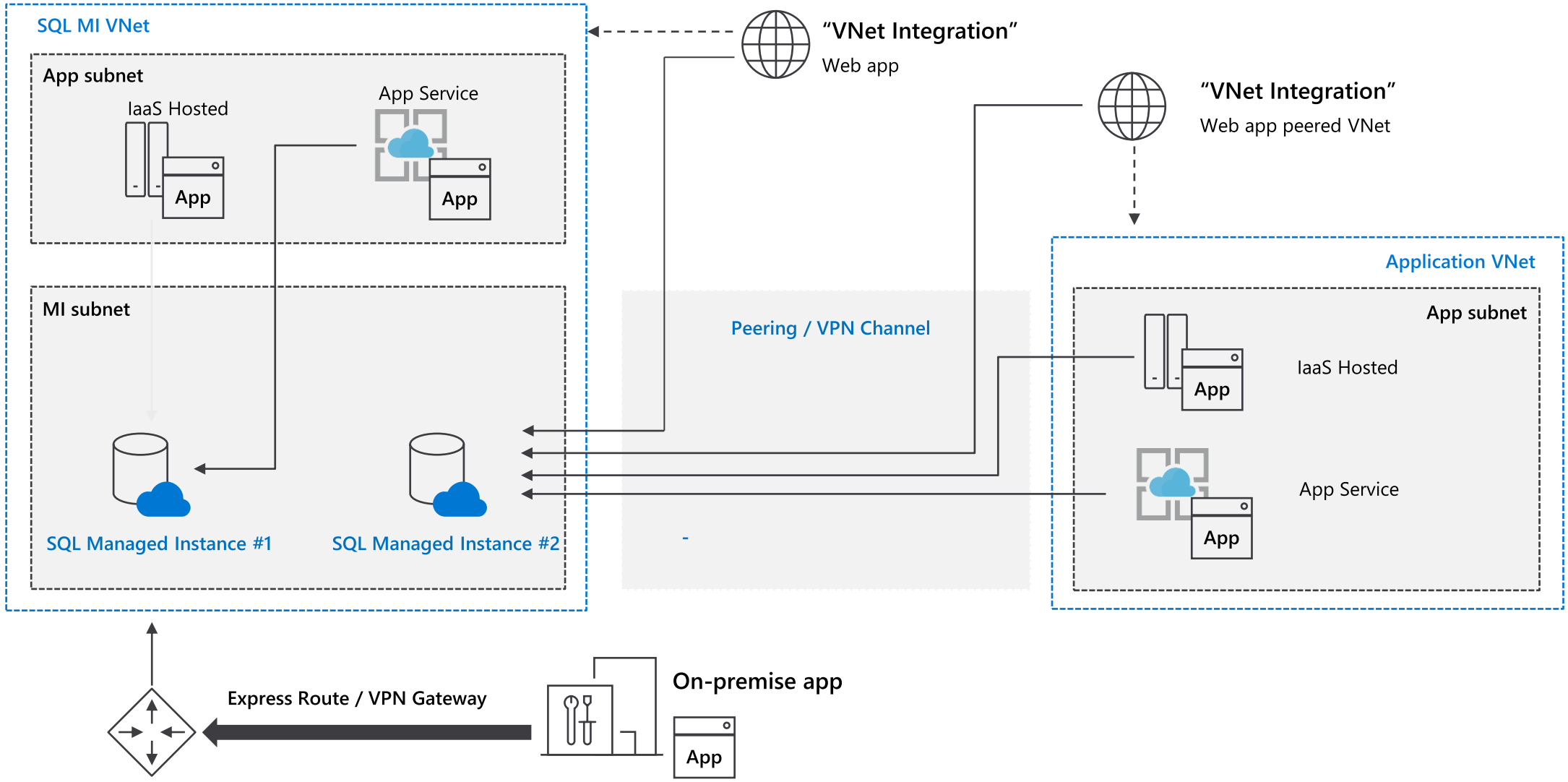
Isolation and connectivity of Managed Instance



MI Virtual Cluster



Host your application in the cloud or keep on-premises



Virtual network considerations

Be empty: The subnet must not contain any other cloud service associated to it, and it must not be Gateway subnet. You won't be able to create Managed Instance in subnet that contains resources other than managed instance or add other resources inside the subnet later.

Have specific route table: The subnet must have a User Defined Routes to Microsoft Public IP Addresses

Optional custom DNS: If custom DNS is specified on the VNet, Azure's recursive resolvers IP address (such as 168.63.129.16) must be added to the list.

No Service endpoint: The subnet must not have a Service endpoint (Storage or Sql) associated to it. Make sure that Service Endpoints option is Disabled when creating VNet.

Sufficient IP addresses: The subnet must have minimum of 16 IP addresses. For more information. By design, a Managed Instance needs a minimum of 16 IP addresses in a subnet and may use up to 256 IP addresses. As a result, you can use subnet masks /28 to /24 when defining your subnet IP ranges.

Azure uses five IP addresses in the subnet for its own needs

Each General Purpose instance needs **two** addresses

Each Business Critical instance needs **four** addresses

Reference docs for latest info: https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance-vnet-configuration#requirements

Virtual network guidance

A Managed Instance must be deployed in an Azure Virtual Network

Allows for connecting directly from an on-premises network

Allows for connecting linked servers or other on-premises data stores

Allows for connecting to additional Azure resources

Plan your deployment

Managed Instance requires a minimum of 16 IP addresses in a subnet and may use up to 256 IP addresses

If deploying multiple Managed Instances inside the subnet, you need to optimize the subnet size

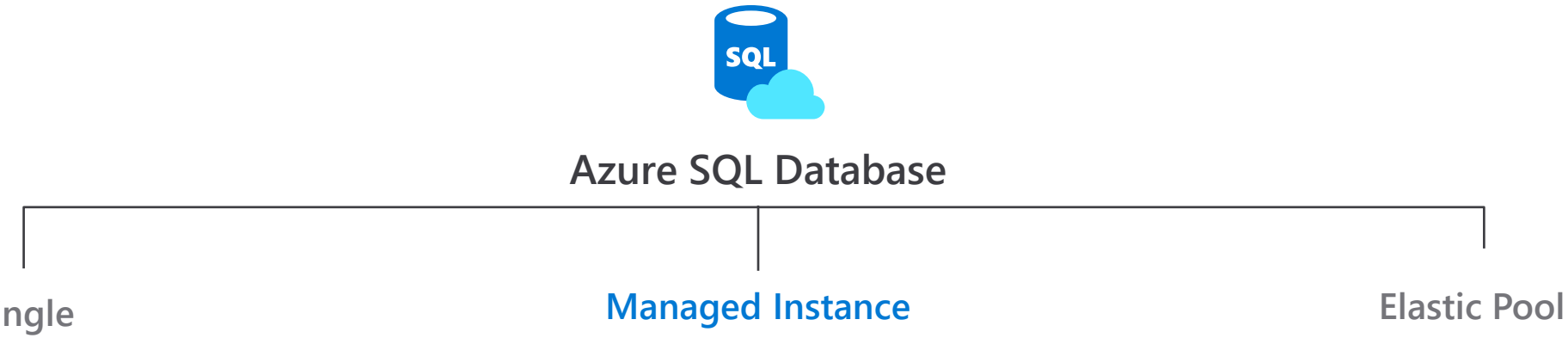
The default values create a subnet that takes all the VNet address space, allowing for only Managed Instance inside the virtual network

Routes

Effective routes on the Managed Instance subnet are not supported

Routes can be user-defined (UDR) or Border Gateway Protocol (BGP) routes propagated to network interfaces through ExpressRoute or site-to-site VPN connections

Managed Instance key capabilities



Fully-managed service

- Built on the same infrastructure as SQL Database
- Provides the same benefits (PaaS)

SQL Server compatibility

- Fully-fledged SQL instance with nearly 100% compat with on-premise

Full isolation and security

- Contained within your VNet
- Private IP addresses
- Express Route / VPN connectivity

New pricing options

- Transparent
- Frictionless
- Competitive

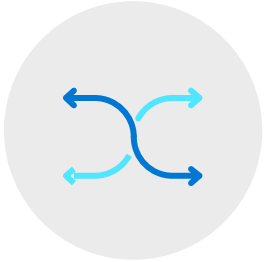
SQL Server compatibility

Familiarity

Leverage SQL Server skills across on-premises and cloud environments with a familiar relational foundation and T-SQL functions, including spatial data support for location-based apps.

Tools

Choice of management tools: APIs, Azure Management Portal with HTML5 support, SQL Server Management Studio (SSMS) or Azure Data Studio.



Flexibility

Support seamless development online, offline, and across on-premises and cloud-designed apps with Visual Studio. Extend existing applications to the cloud with DAC framework support.

Broader SQL Server support for improved compatibility on Azure

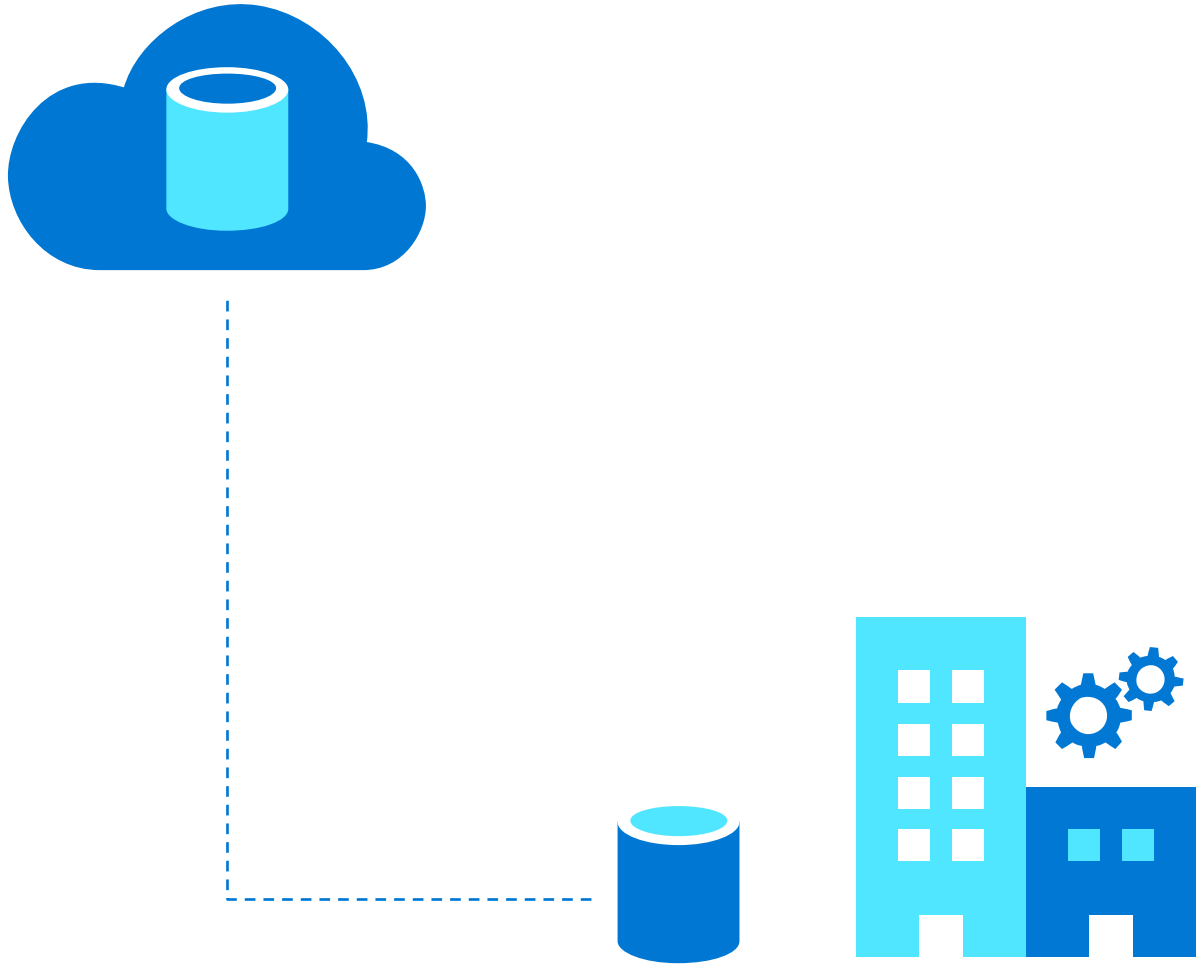
Online index rebuild capability for clustered and non-clustered indexes for greater availability

Build highly optimized schemas to improve query processing with table partitioning support

Access Common Language Runtime (CLR) and define CLR types, aggregates, functions, and procedures written in C#

In-Memory Columnstore index for data marts

Support for additional Dynamic Management Views (DMVs) for deeper insight into application health



Scale your data workloads with Azure SQL Database

Challenges with managing Very Large Databases (VLDB)

Size of data

Operations take a LONG time (days in some cases)

Ongoing operations degrade database performance

Can cause outages and downtime

Provisioning more storage to expand the database can be painful

Scaling Compute

Logistics of moving to larger box

Economics of sizing for max peaks

Hyperscale is the foundation for massive app growth

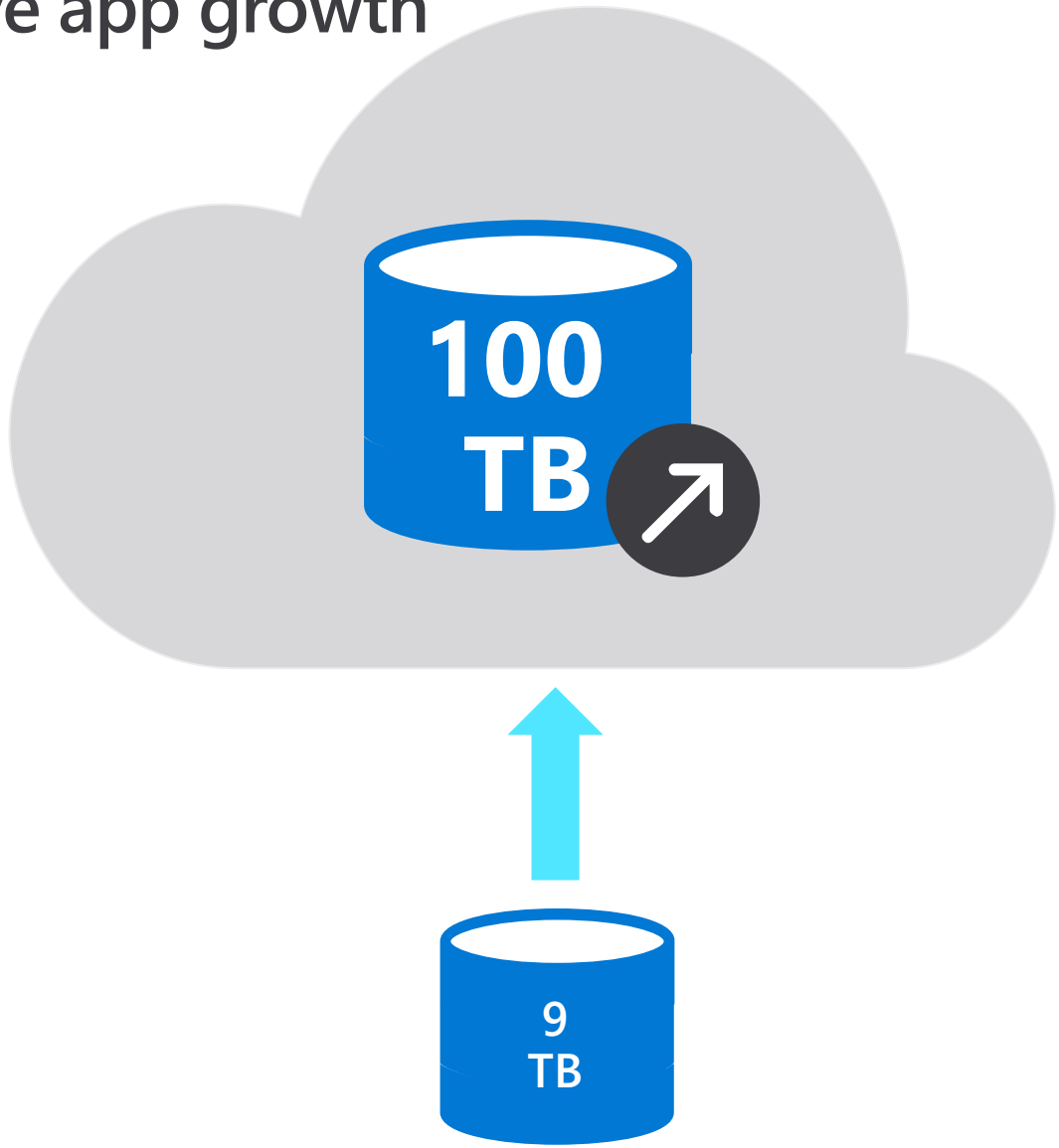
Hyperscale is a new, highly scalable service tier that adapts on-demand to your workload's needs, auto-scaling up to 100TB per database.

Storage dynamically adapts to your workloads' needs, auto-scaling up to 100TB.

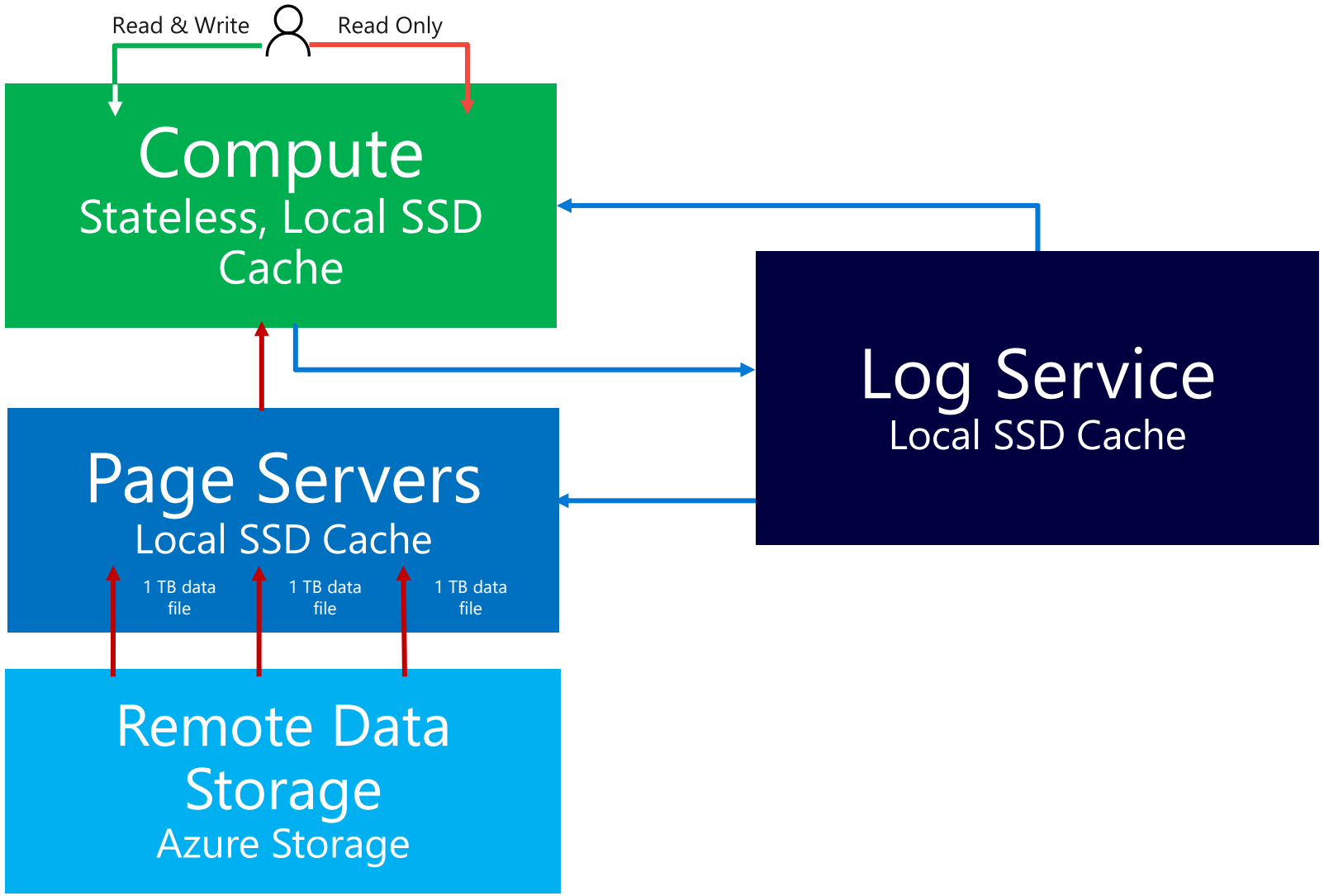
Provision one or more additional compute nodes that can serve your read-only workload and use them as a hot-standby, in case of failover.

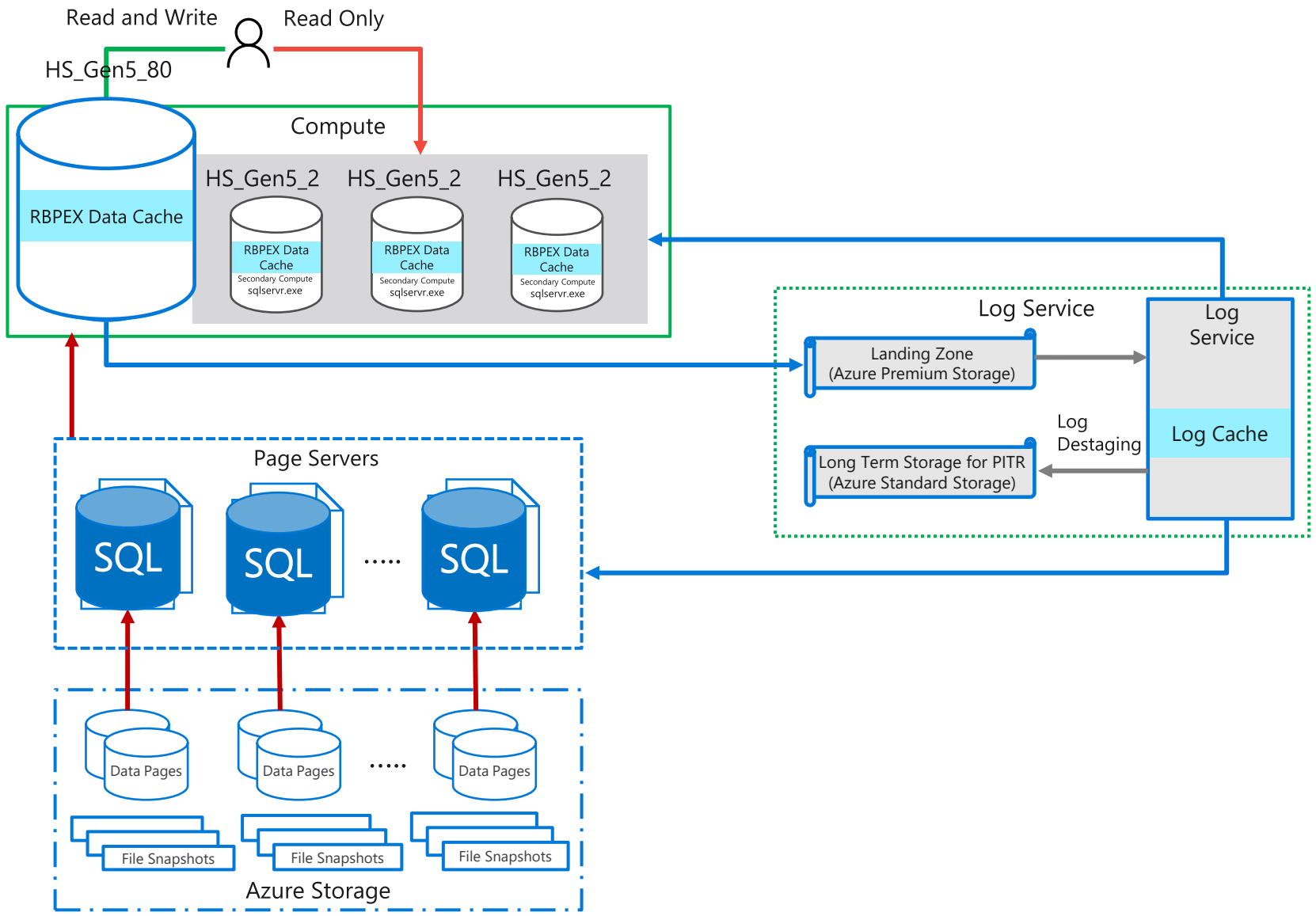
Perform operations in constant time, regardless of the size of the data operation.

Compute and storage resources scale rapidly and independently without sacrificing performance.



Hyperscale components





Constant time scale up and down

Offload read-only workload by adding read-scale replicas without data copy – constant time as well

Low log commit latency - <2.5ms with Premium Storage; < 0.5ms with Ultra SSD (future improvement)

High log generation rate and fast data loading

Page server instances work independently – infinite scale out

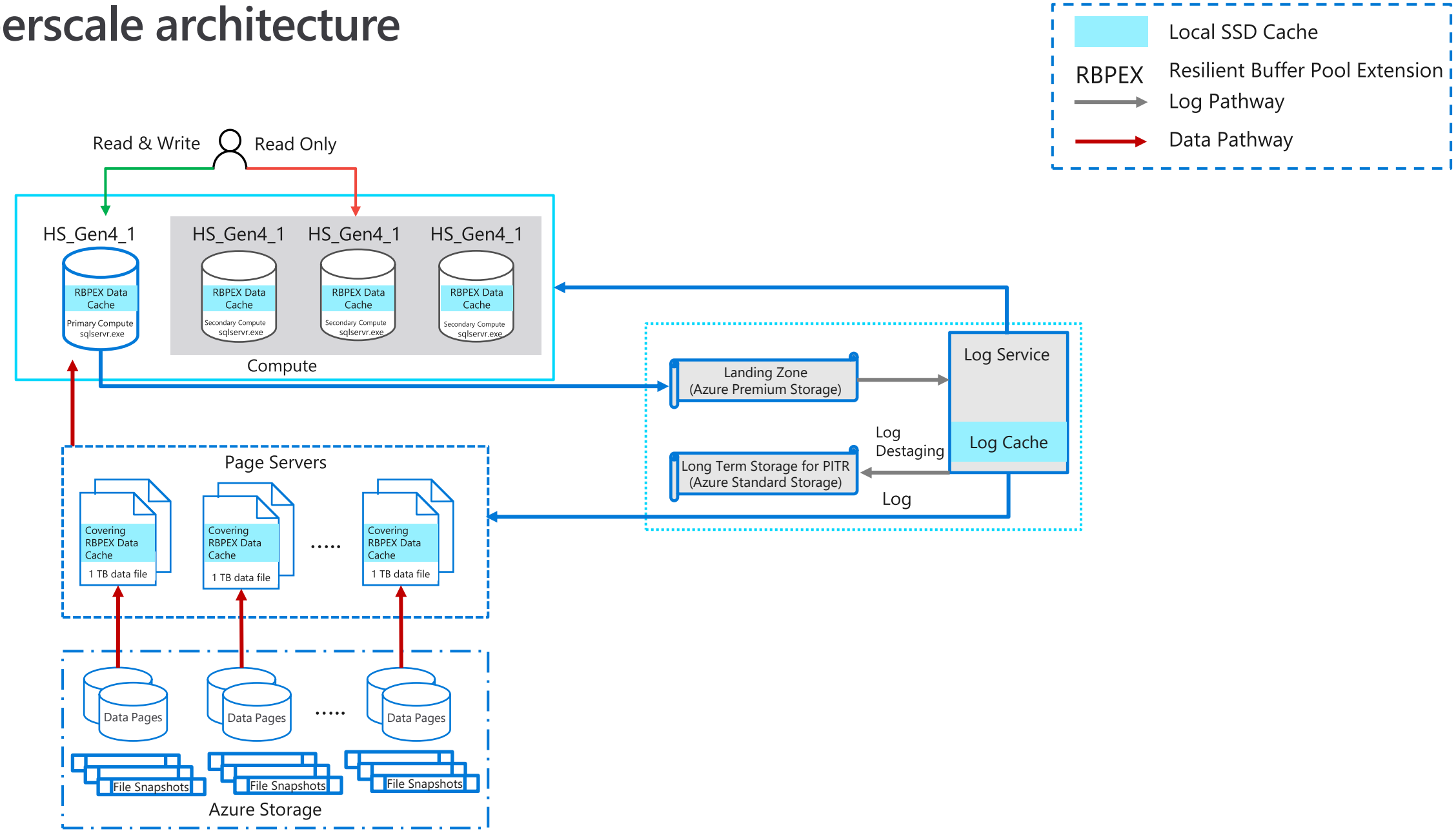
Adding more page servers as your database grows – pay by usage (start with 5GB with 1GB increments)

Snapshot backup + log offloading – zero impact to compute resource

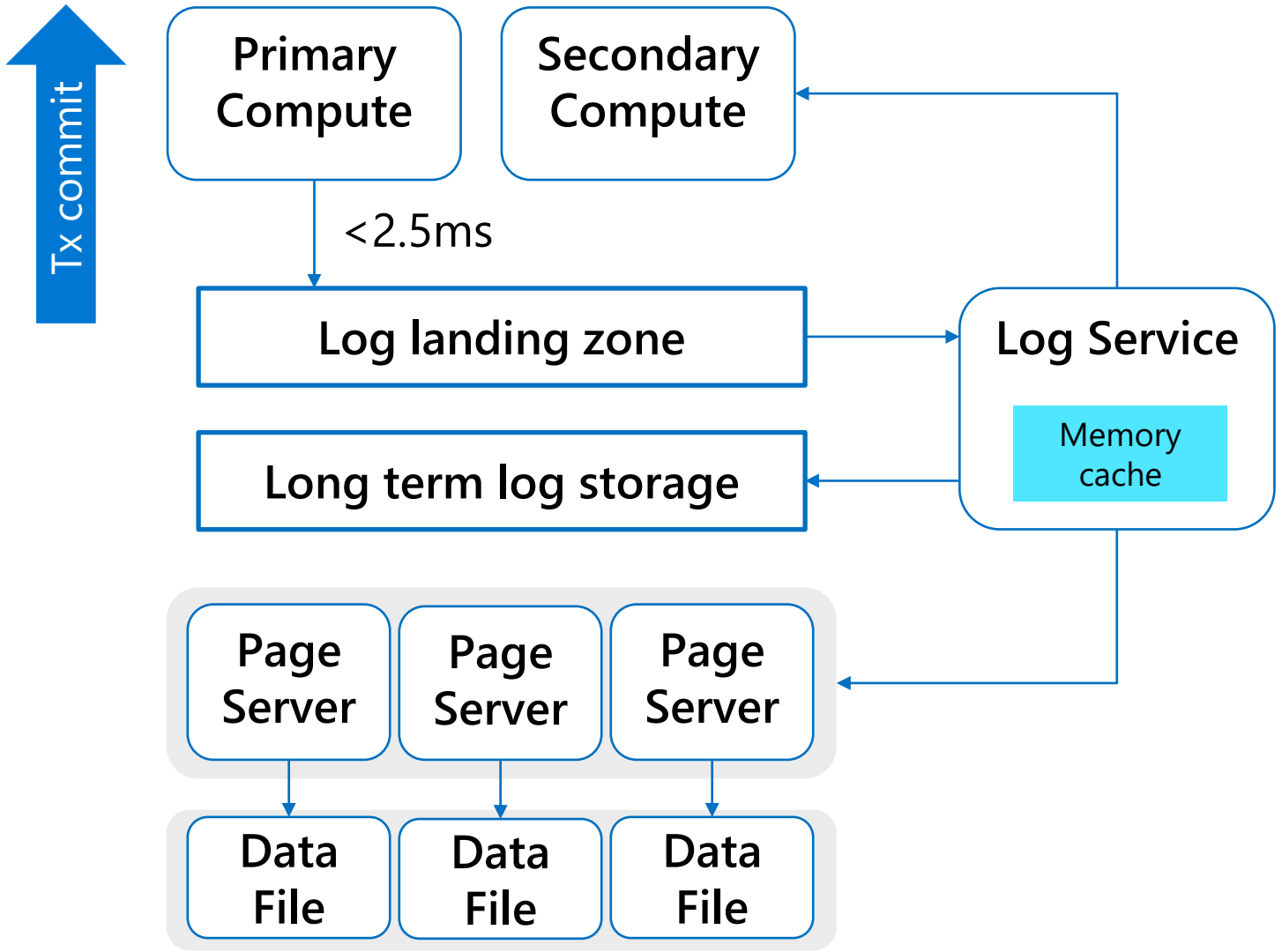
Restore database by copying snapshots and log records – constant time point in time restore

Checkpoints also offloaded to page servers

Hyperscale architecture



Write IO



Primary compute write log to the log landing zone - <2.5ms latency

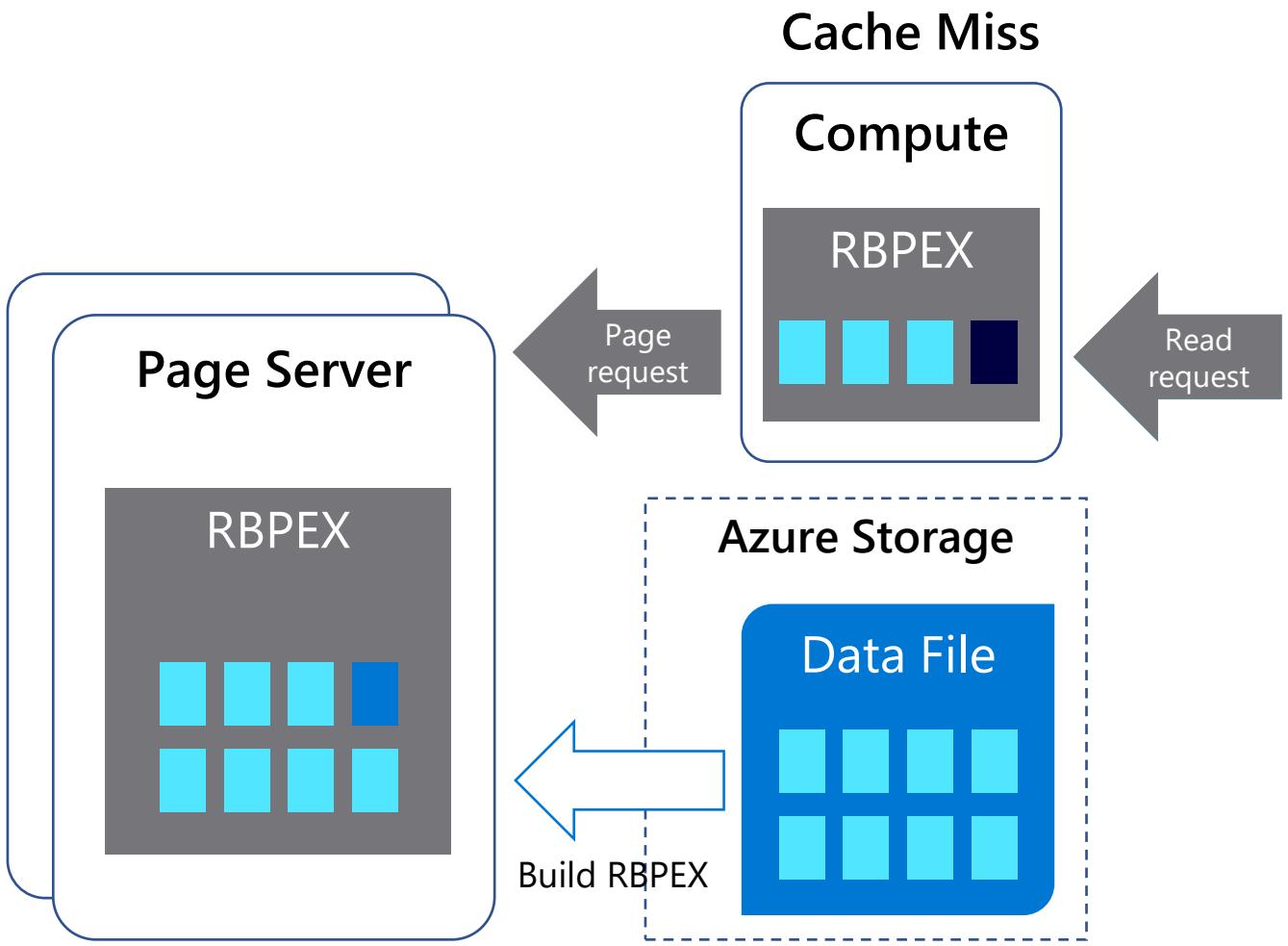
Commit transactions after hardening the log

Async log apply to the secondary compute

Async log apply to page servers

Page server writes data to data files at checkpoints

Read IO



Pre-build RBPEX when page server instance started

Two page-server replicas guarantee IO latency

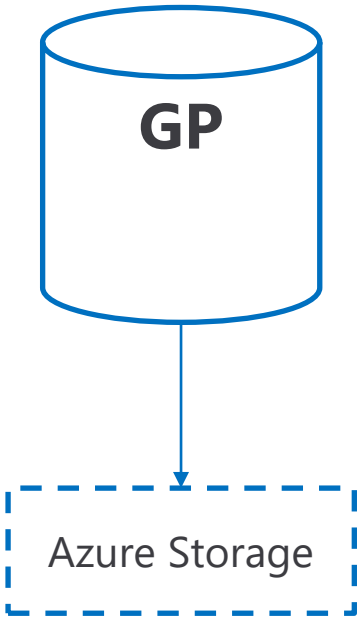
RBPEX on compute nodes is proportional to # of vCores

Hit local RBPEX - < 0.5ms
Hit page server RBPEX - < 2ms

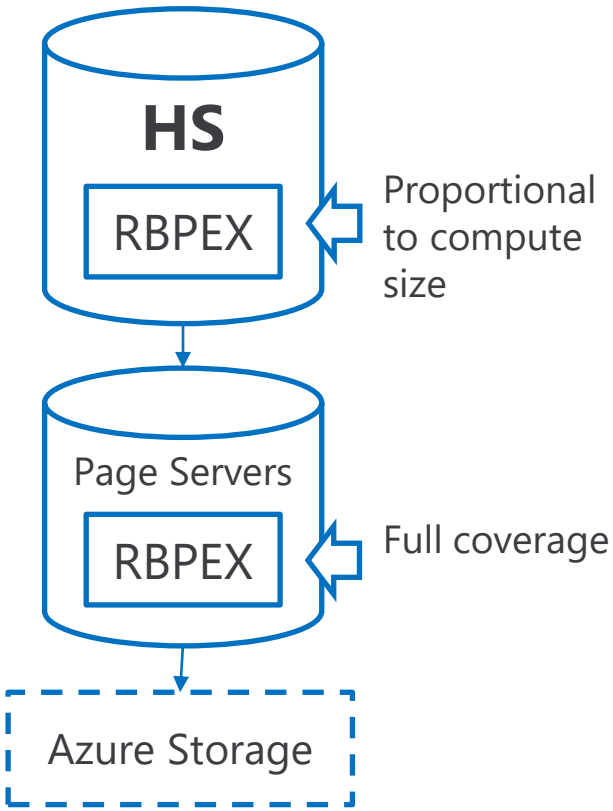
Optimized for OLTP workload – operating on hot data

Use Column Store index to optimize HTAP/OLAP workload

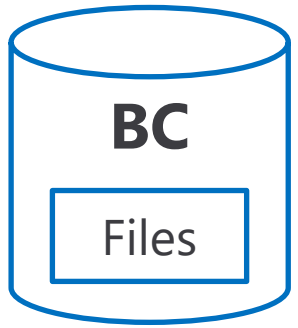
Read IO



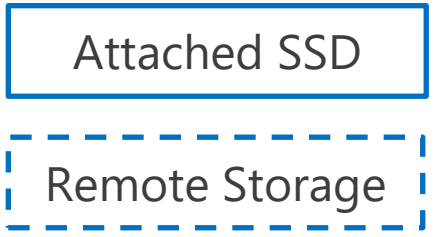
~2ms for all data access



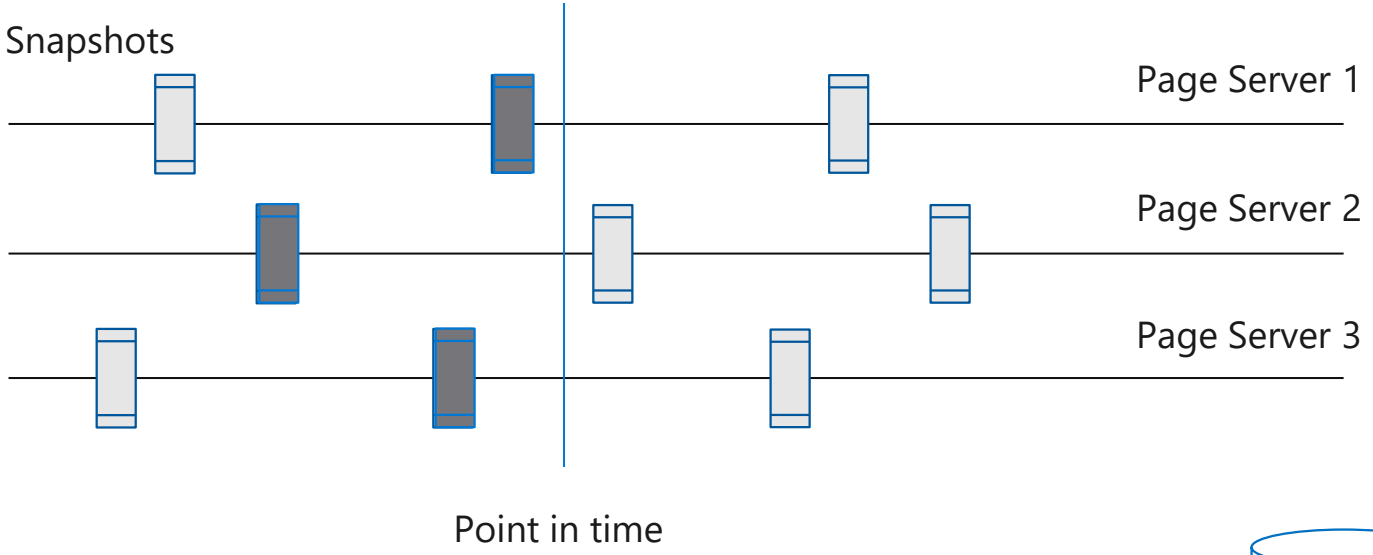
<0.5ms for hot data access
~2ms otherwise

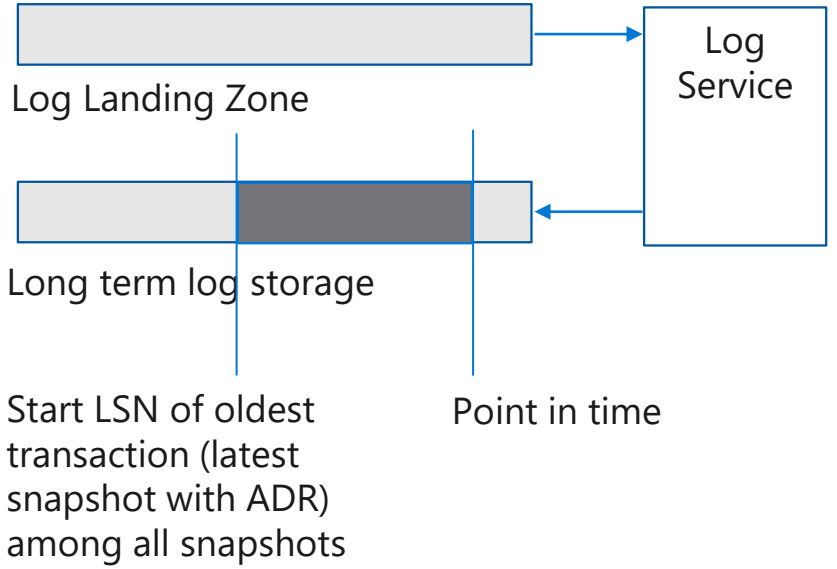


<0.5ms for all data access



Backup & Restore



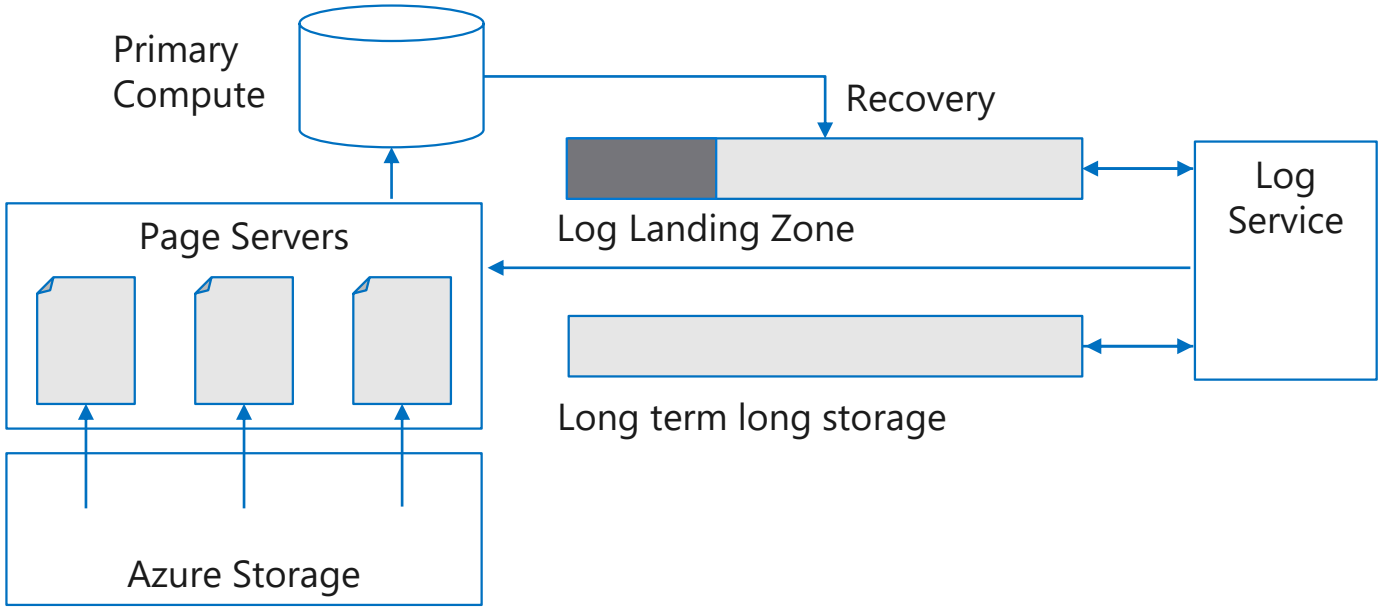


Backup has no impact to apps

Parallelized copy (metadata)

Constant time PITR

Accelerated database recovery



Scaling multiple databases across shared resources with elastic pools

Elastic database model

Elastic databases in elastic database pools

Pooled resources are used by many databases

Standard elastic database pools provide 50-3000 database throughput units (DTUs) for up to 500 databases

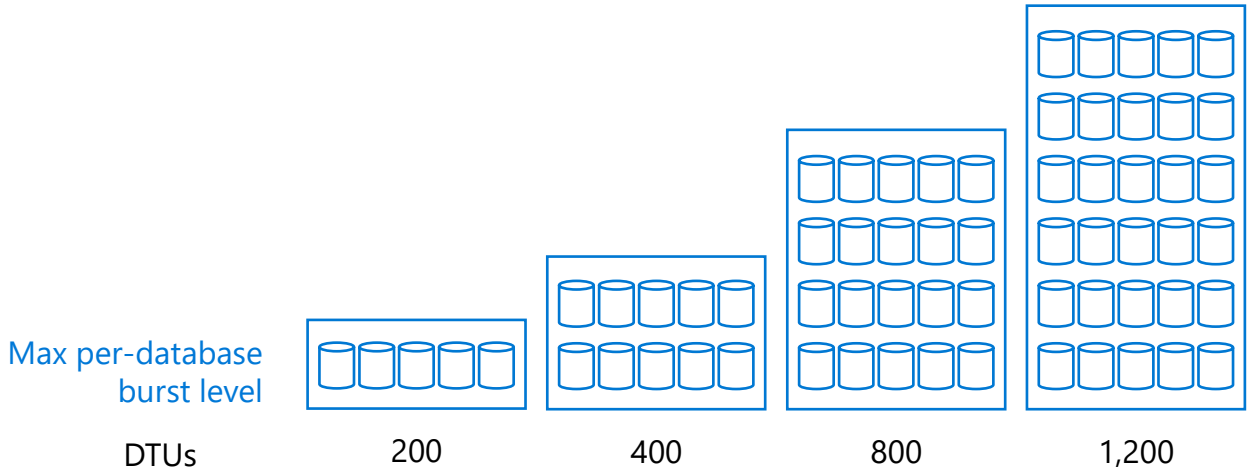
Max eDTUs per database can be set if available based on utilization by other database in the pool

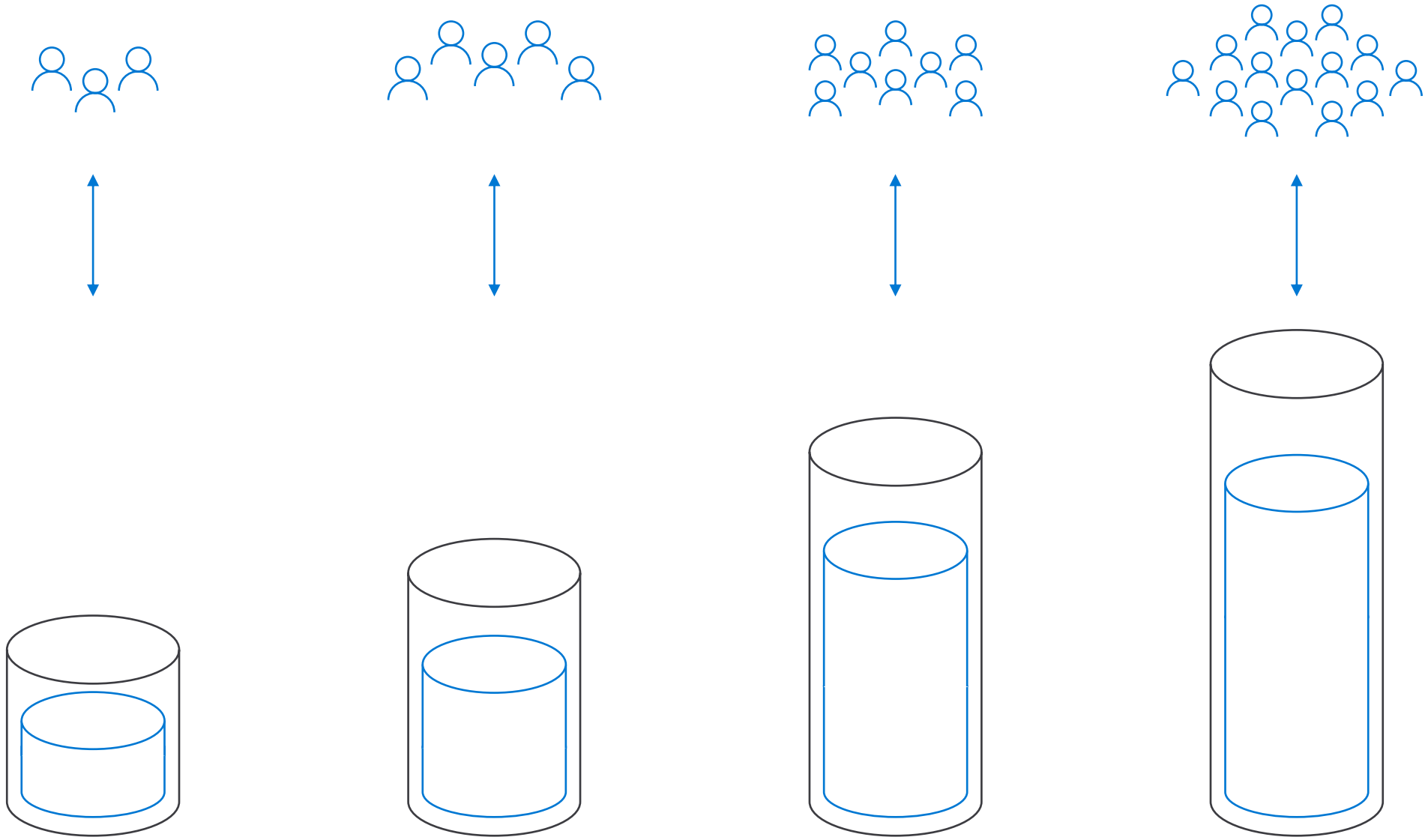
Create/configure pools using portal, Azure PowerShell, REST APIs

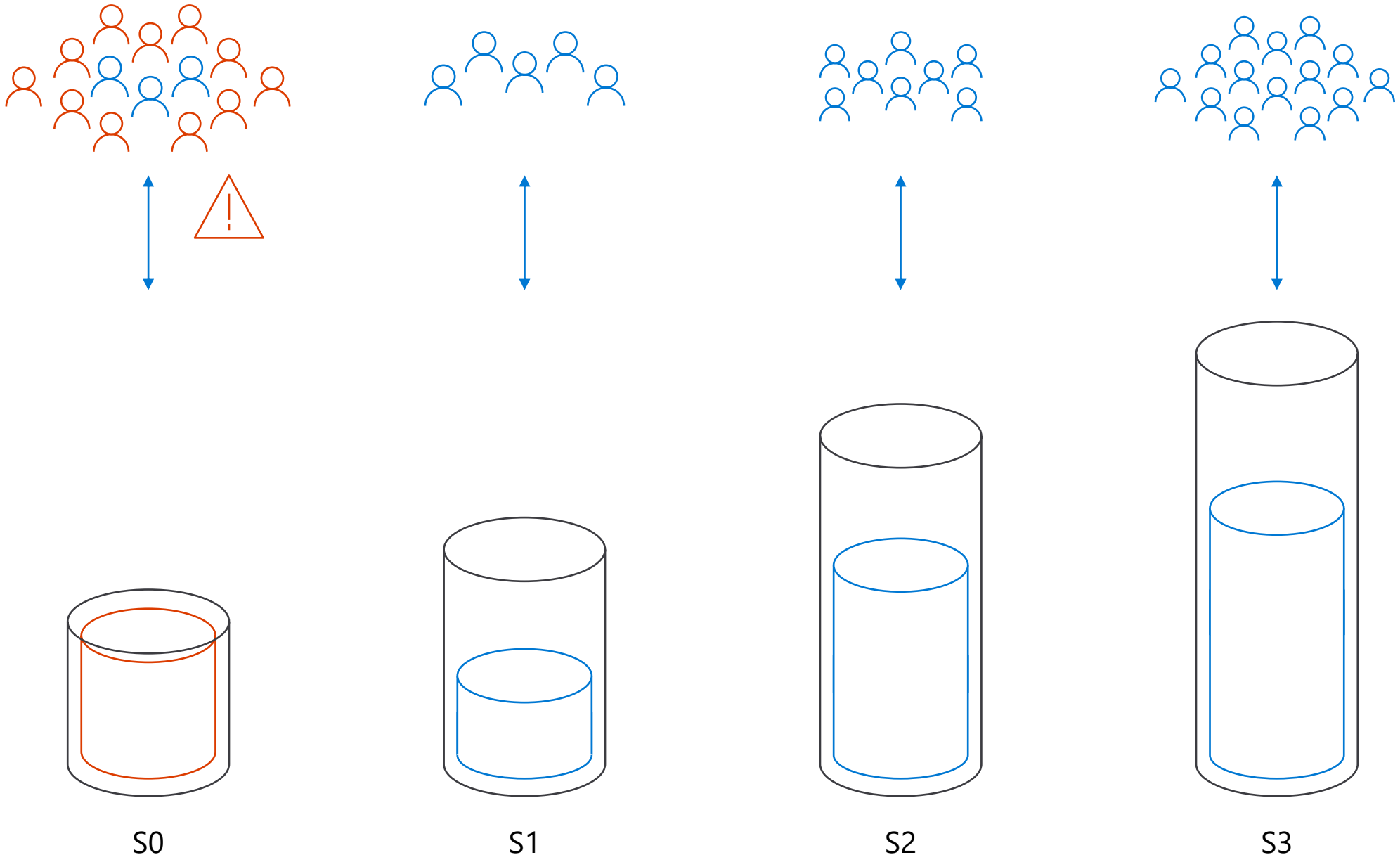
Move databases in/out using portal, Azure PowerShell, REST APIs, and T-SQL

Databases remain online throughout

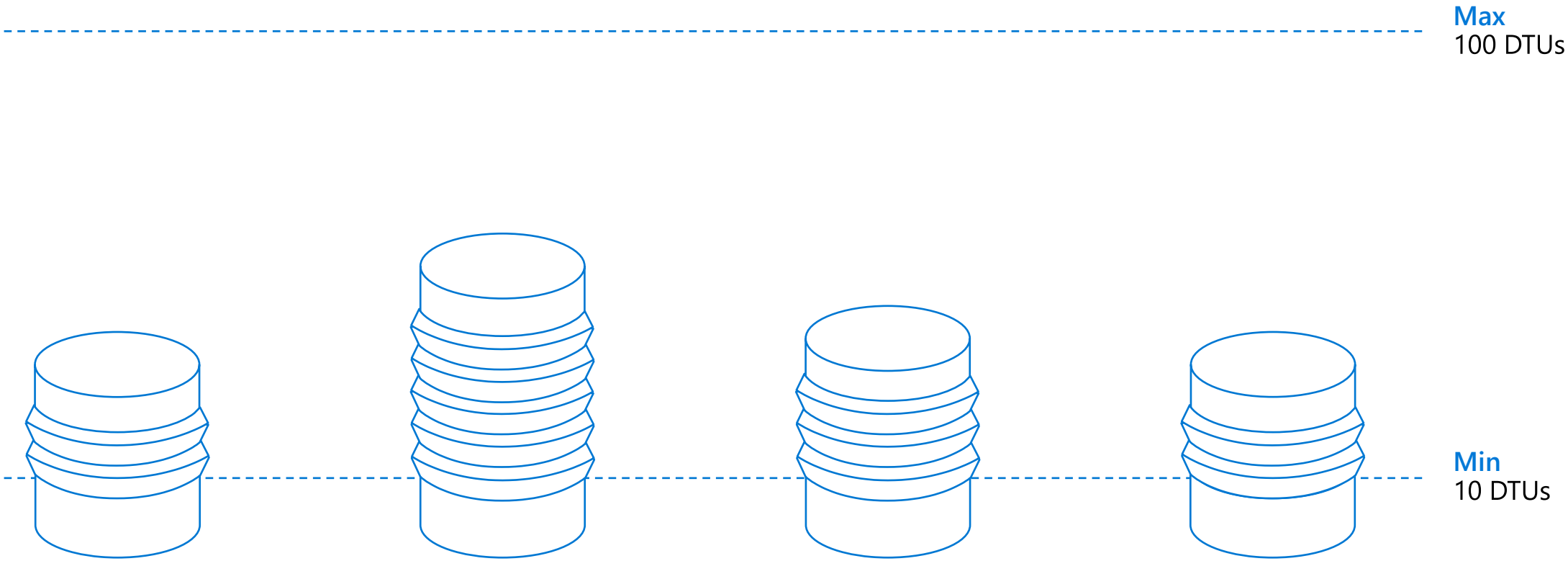
Monitoring and alerting available on both pools and databases




S0
S1
S2
S3


S0
S1
S2
S3

Elastic database pool

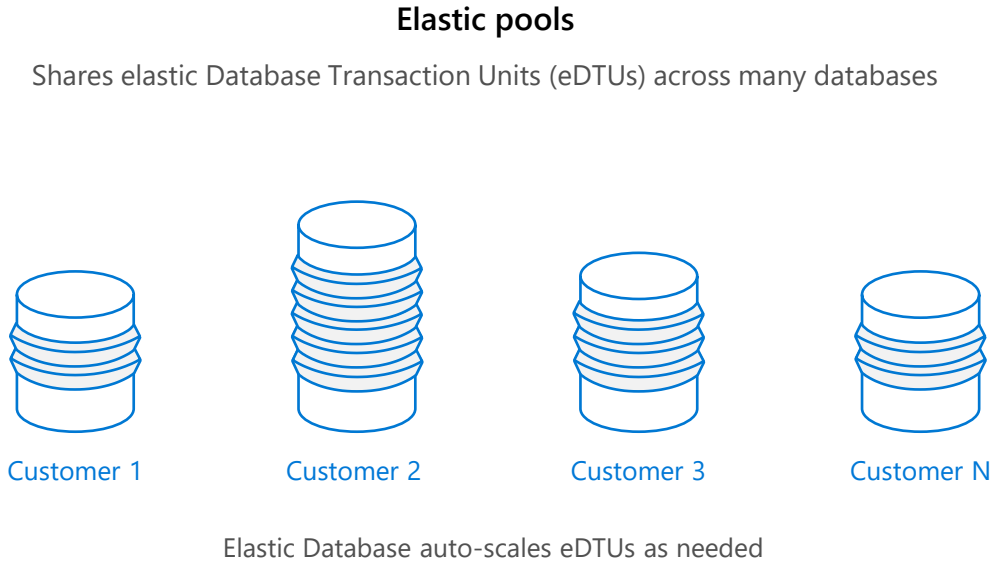


Auto-scaling you control with Elastic Database

Pools automatically scale performance and storage capacity for elastic databases—anytime, anywhere

Control the performance assigned to a pool, add or remove elastic databases on demand, and define performance of elastic databases without effecting overall pool cost

Don’t worry about managing usage needs of individual databases



Elastic database pool service tiers

Buy a fixed number of eDTUs, share compute across many databases

ELASTIC DATABASE POOLS

Manage operational activities across multiple databases

Elastic database tools

- Elastic database jobs
- Elastic database queries
- Elastic database transactions

Support management and increased efficiency for multi-database environments

Elastic database pools

Elastic database jobs overview

Elastic database jobs enables you to run T-SQL scripts (jobs) against all databases in an elastic database pool

Define, maintain, and persist T-SQL scripts to be executed across an elastic database pool

Execute T-SQL scripts reliably with automatic retry and at scale

Track job execution state

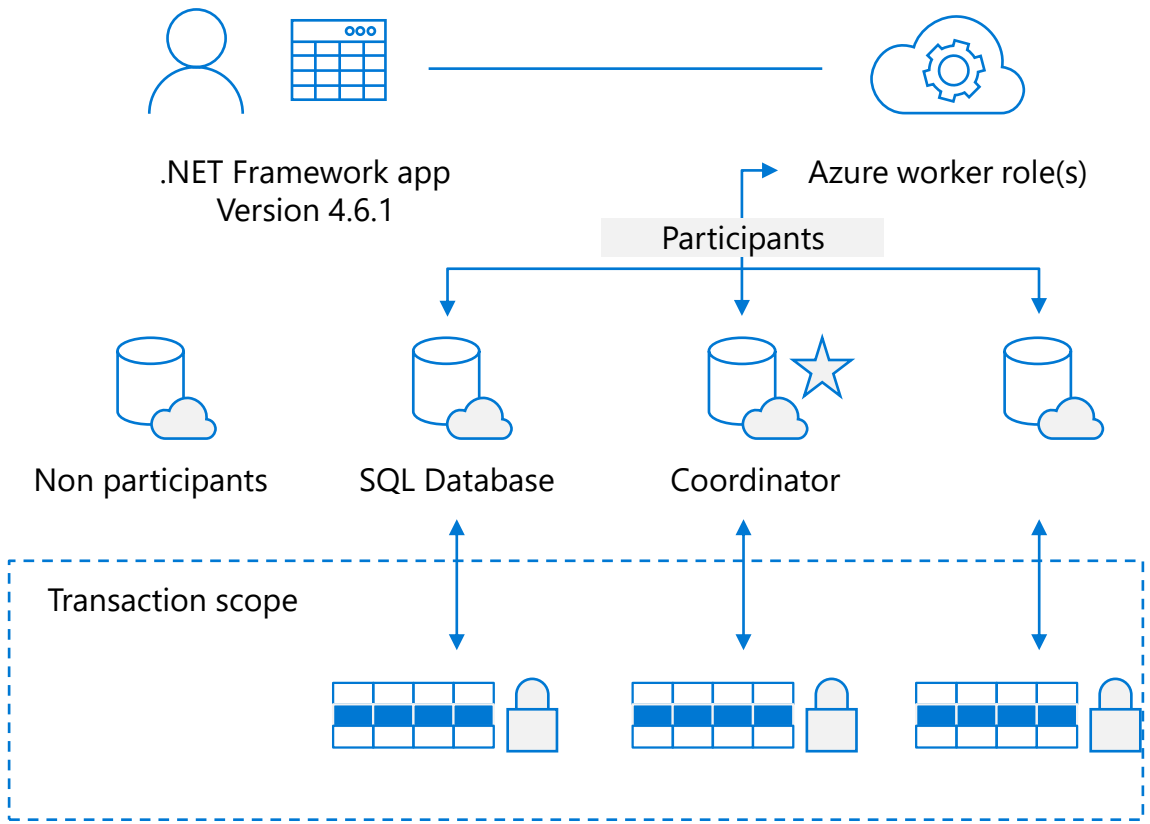
Elastic database transactions

Distributed transactions: perform operations across several databases with transactional properties in Azure SQL Database

Ensure that changes are made to all databases or not at all

Use the same .NET APIs that are used on-premises today

Distributed transactions on Azure SQL Database



Querying across many databases

Connect to a single Azure SQL Database instance using familiar Azure SQL Database connection strings

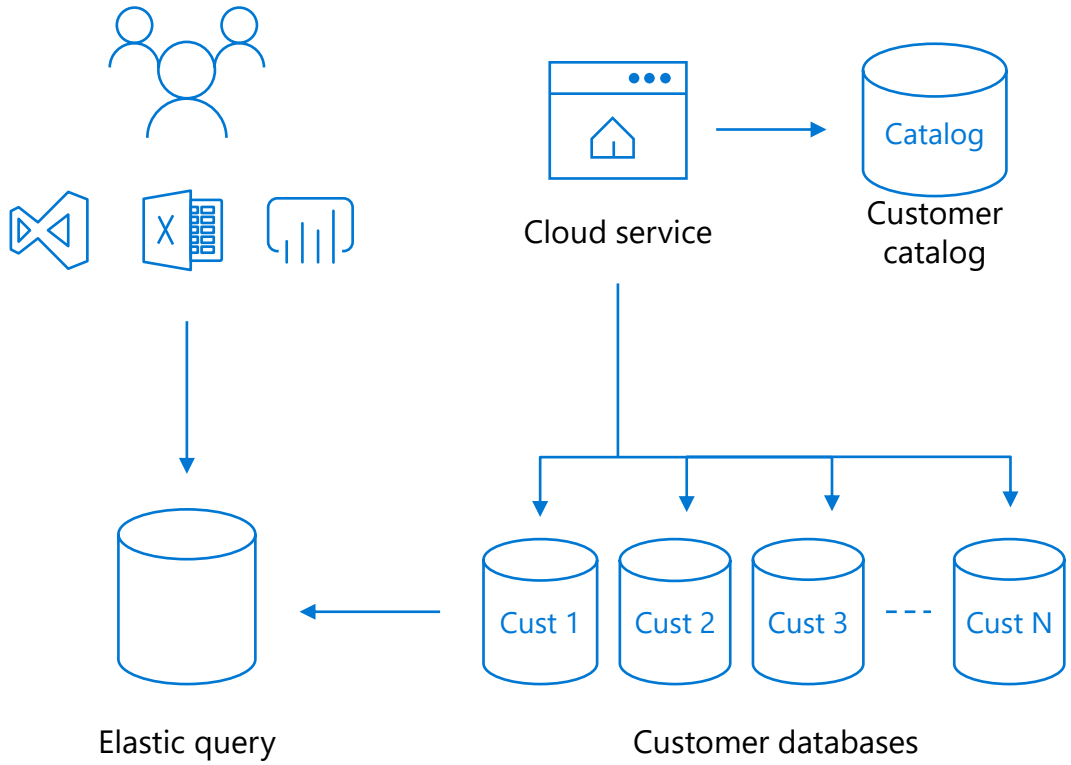
Simplify setup with Transact-SQL Data Definition Language (DDL)

Transparently query many databases from a single database

Familiar programming experience with Transact-SQL, ADO .NET LINQ, Entity Framework (EF), and others

Use familiar development and business-intelligence (BI) tools to query across many databases

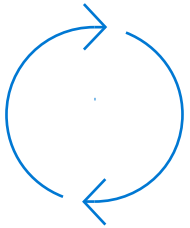
Designed for interactive querying



Multiply capacity and density for scalability

Adding more capacity

Identifying and breaking contention and choke points
How to add additional capacity to a solution?
Subtle constraints to consider…



Using capacity more efficiently

Traditional performance tuning
Maximizing application throughput (for example, leveraging batching)
Improving network performance

Elastic database client library overview

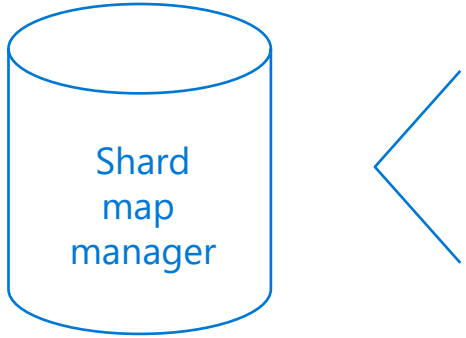
Two types of shard maps

Range: contiguous values

List: explicit values

Four types of sharding keys

INT, BIGINT, GUID, VARBINARY



[shardmaps_global]

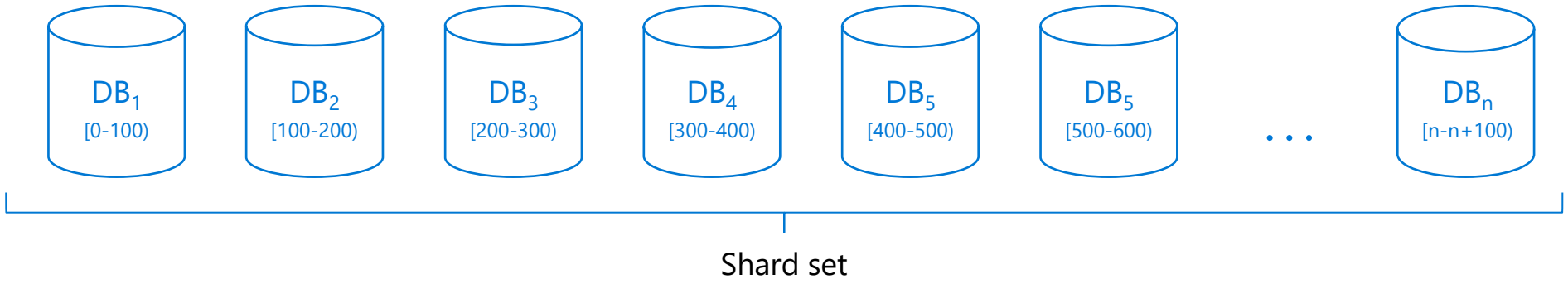
smid	name
1	RangeShardMap

[shards_global]

sid	smid	Datasource	Databasename
1	1	serverName	DB2
2	1	serverName	DB2

[shard_mappings_global]

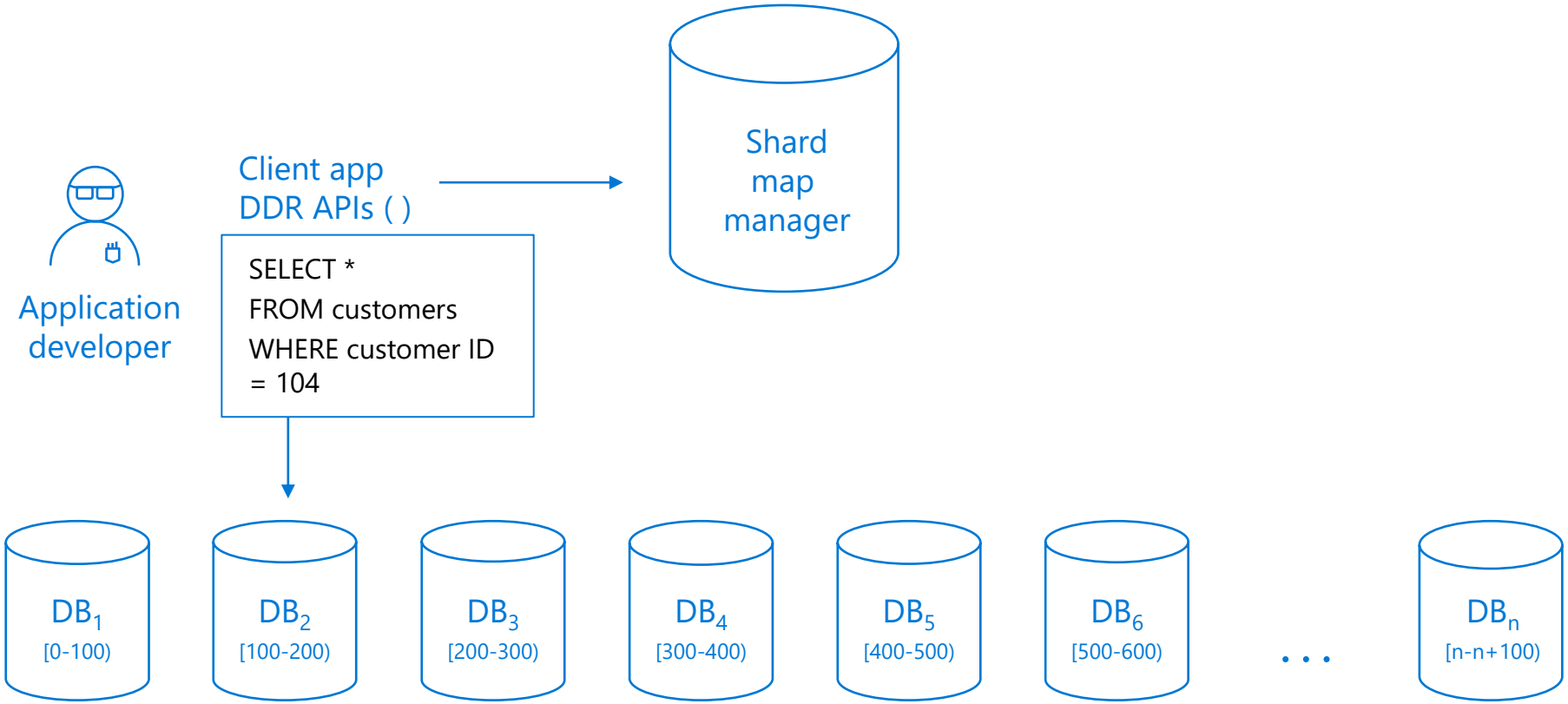
mid	smid	min	max	Sid
1	1	0	100	1
2	1	100	200	2



Data Dependent Routing (DDR)

Scenario:

Query a shard with a specific shardlet key



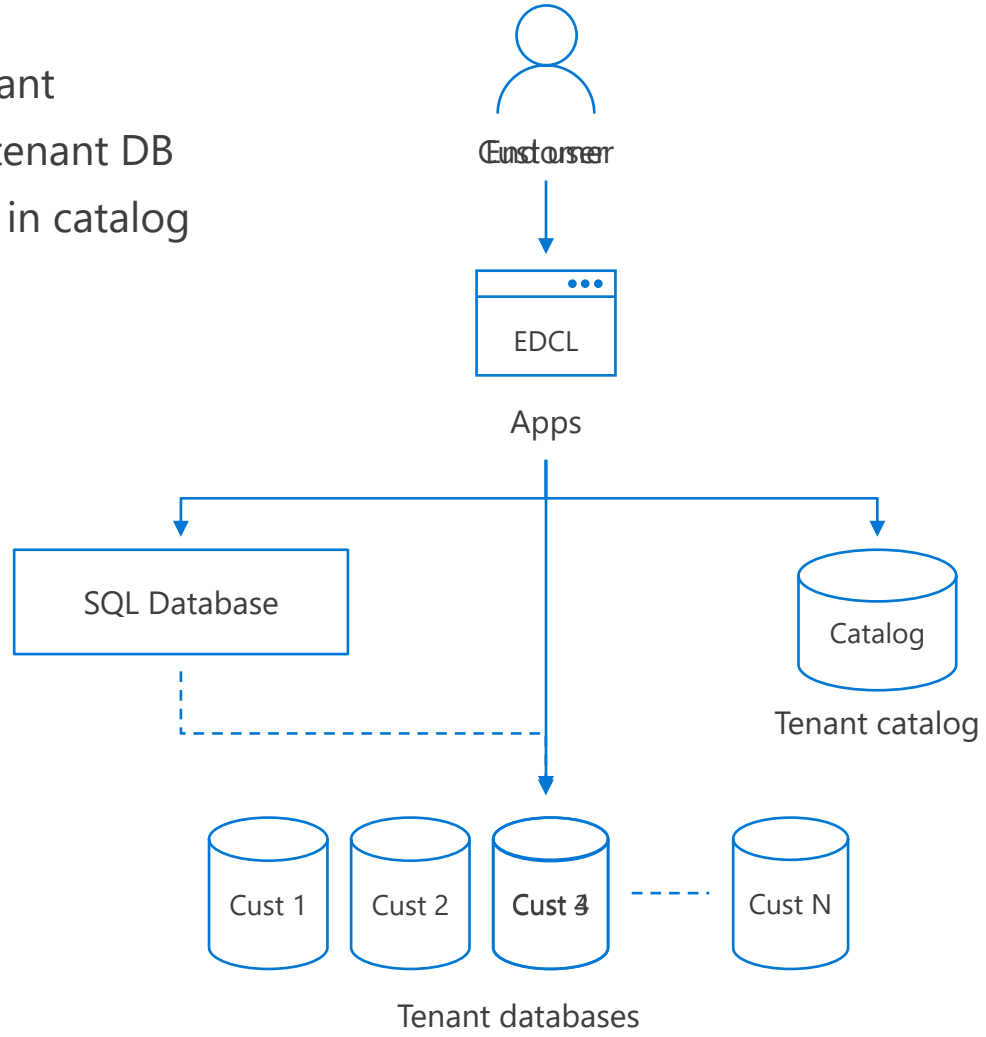
Data Dependent Routing (DDR)

```
// Get a routed connection for a given shardingKey
using (SqlConnection conn = ShardMap.OpenConnectionForKey(
                    shardingKey,
                    connectionString /* Credentials Only */ ,
                    ConnectionOptions.Validate   /* Validate */ ));
{
     using (SqlCommand cmd = new SqlCommand()
     {
          cmd.Connection = conn;
          cmd.CommandText = "SELECT dbNameField, TestIntField, TestBigIntField FROM
ShardedTable";

          SqlDataReader sdr = cmd.ExecuteReader();

          // Now consume results from the data reader…
```

Mapping tenants to databases

Tenant onboarding

A. Customer signs up as new tenant
B. Cloud service provisions new tenant DB
C. …registers tenant id, DB name in catalog



Application use

1. End user connects via a front end to the cloud service
2. Cloud service looks up tenant in catalog
3. …connects to correct tenant database
4. …on subsequent requests, uses cached database location

Mapping tenants to databases

Elastic Database Client Library plus catalog schema

Define and manage tenant-to-database mapping

Supports database-per-tenant and sharded data models

Supports split/merge of shards

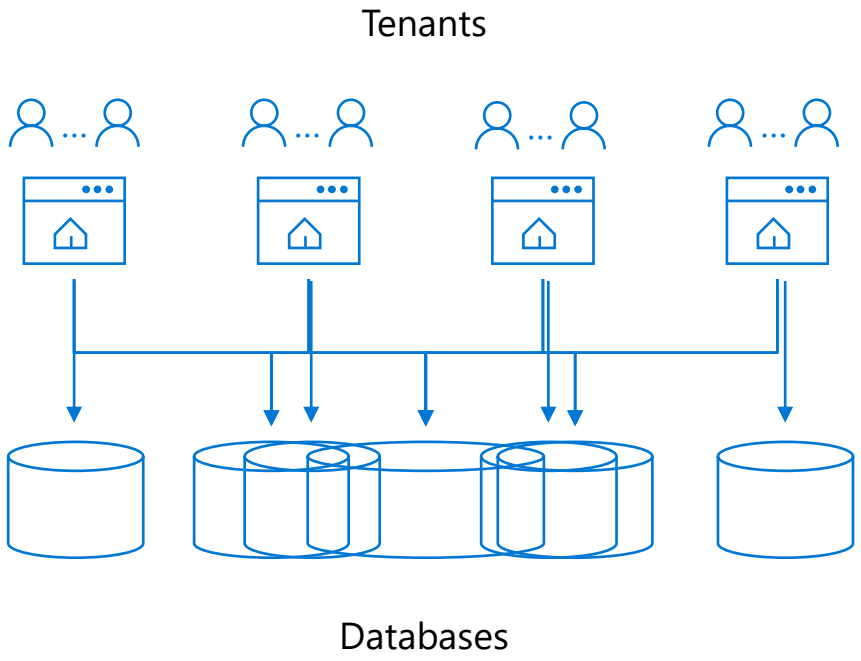
Route database requests (data-dependent routing)

Routes incoming requests to the correct database

Ensures correct routing if databases move or renamed

Caches server and database names to optimize performance

Can be used with native SQL access, EF



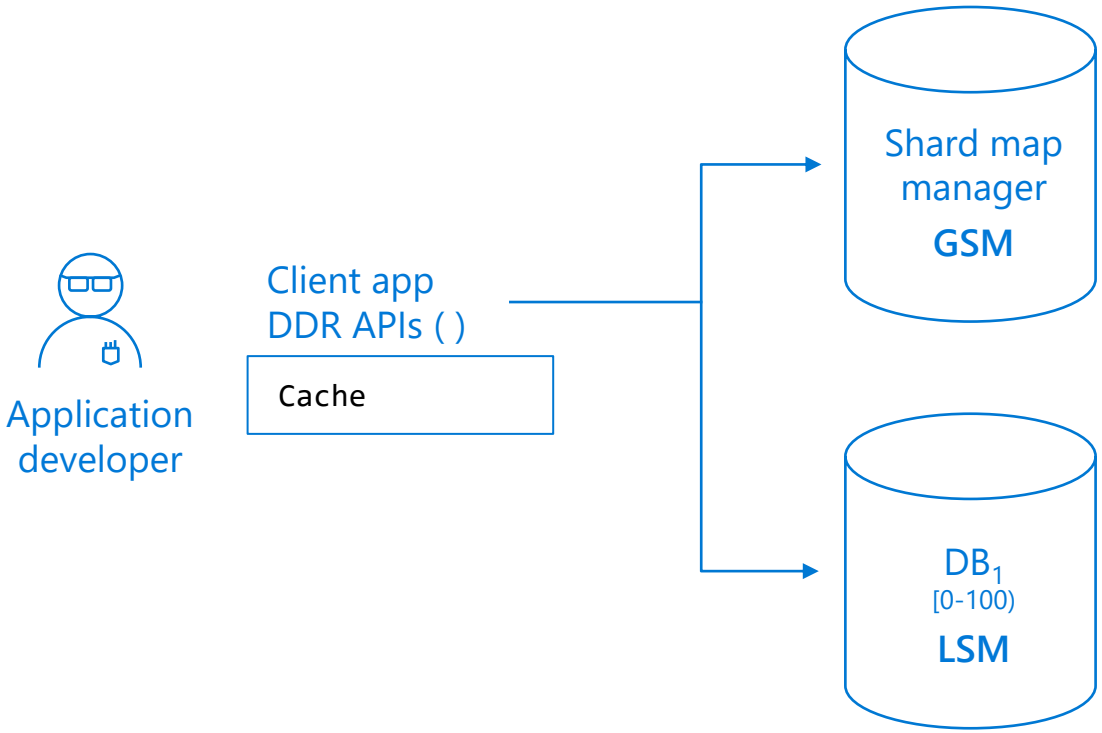
Shard map manager

Caching: Improve performance of shard operations

Global Shard Map (GSM) – state of all shards in the Shard Map

Local Shard Map (LSM) – state of all shards on a particular shard

Client Cache (eager/lazy) – state of all shards in the Shard Map/known shards



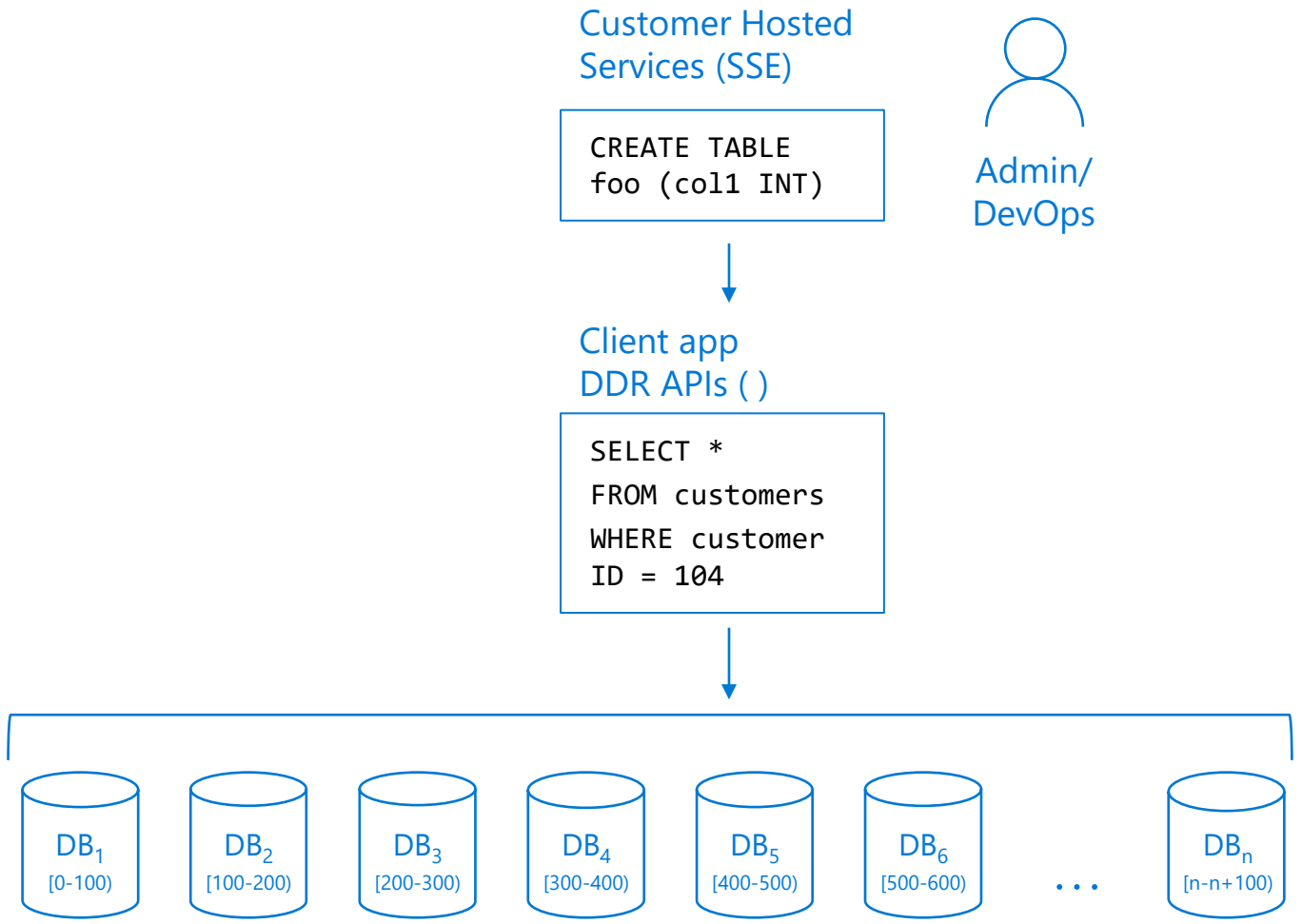
Multi-shard queries (MSQ)

```
using (MultiShardConnection conn = new MultiShardConnection(m_shardMap.GetShards(),
MultiShardTestUtils.GetTestSqlCredential()))
{
    using (MultiShardCommand cmd = conn.CreateCommand())
    {
        cmd.CommandText = "SELECT dbNameField, TestIntField, TestBigIntField FROM ShardedTable";
        cmd.CommandType = CommandType.Text;
        cmd.ExecutionOptions = MultiShardExecutionOptions.IncludeShardNameColumn;
        cmd.ExecutionPolicy = MultiShardExecutionPolicy.PartialResults;

        using (MultiShardDataReader sdr = cmd.ExecuteReader())
        {
            while (sdr.Read())
            {
                var dbNameField = sdr.GetString(0);
                var testIntField = sdr.GetFieldValue<int>(1);
                var testBigIntField = sdr.GetFieldValue<Int64>(2);
                string shardIdPseudoColumn = sdr.GetFieldValue<string>(3);
            }
        }
    }
}
```

Shard Set Executer (SSE)

Scenario:

Perform an admin action across set of shards



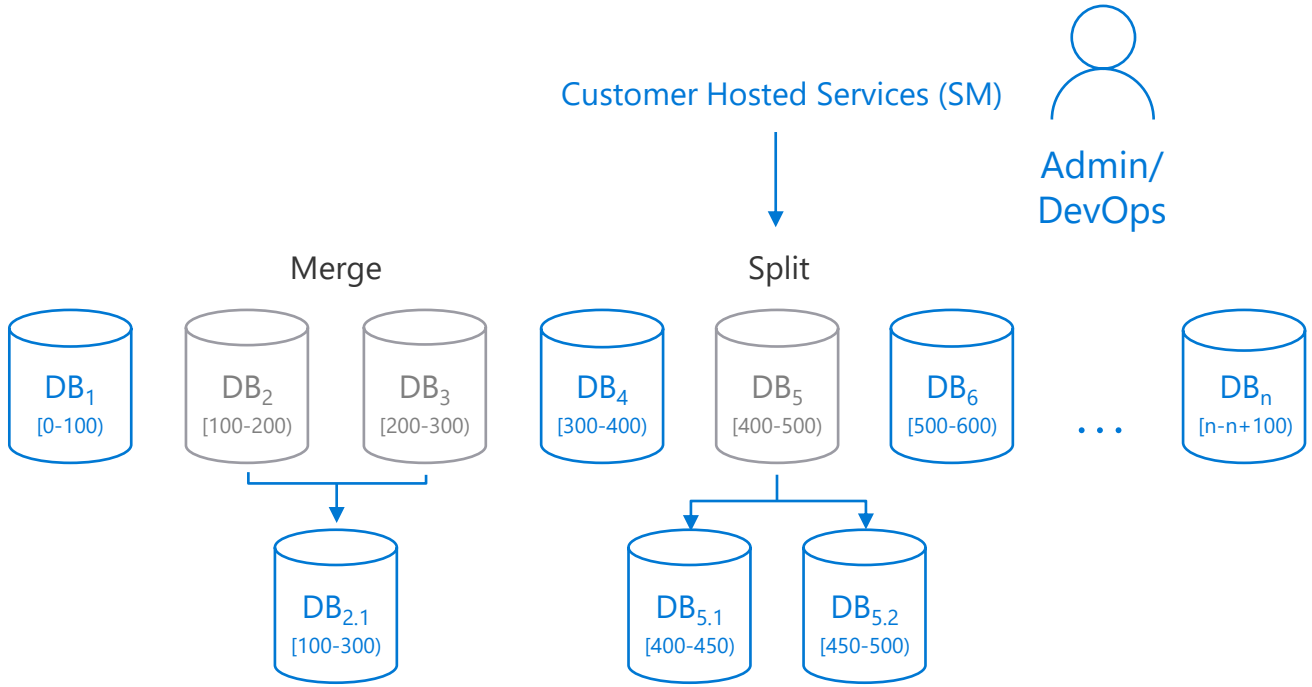
Split/Merge (SM)

Scenario:

Perform a split or merge action

Split: create two distinct shards from one

Merge: create one shard from two distinct shards



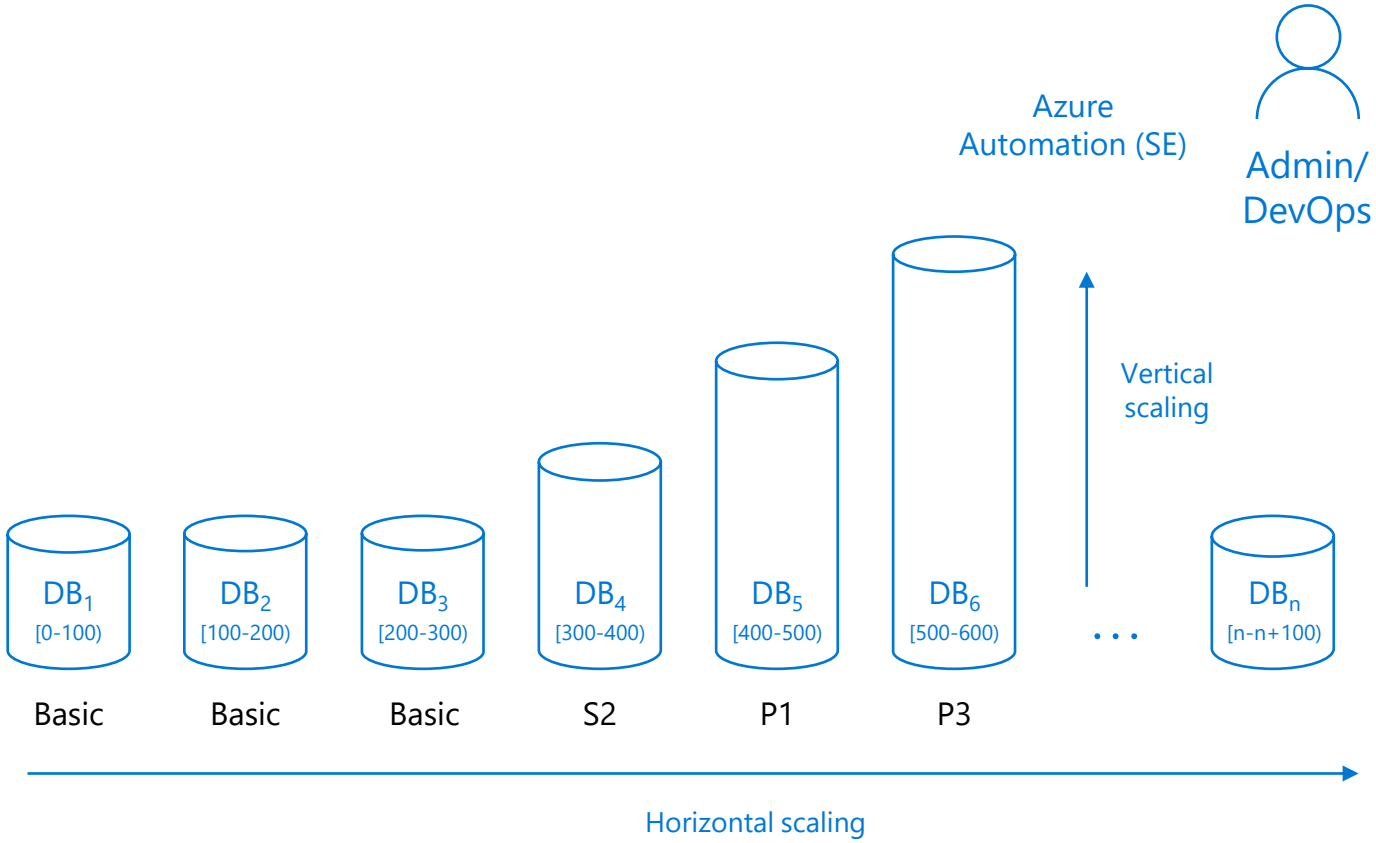
Shard Elasticity (SE)

Scenario:

Automation to vertically scale a shard or horizontally scale a shard set

Vertical scale: increase/decrease the performance level of the shard

Horizontal scale: add/remove a shard to the shard set



The growing need for serverless databases

Why serverless

Compute requirements for new apps may be unknown

Developers struggle to provide sufficient capacity and resources to support apps

Managing unpredictable and intermittent workloads is costly and time-consuming

Businesses struggle to ensure that database provisioning consistently aligns with workload requirements

Existing offerings cannot solve the problem

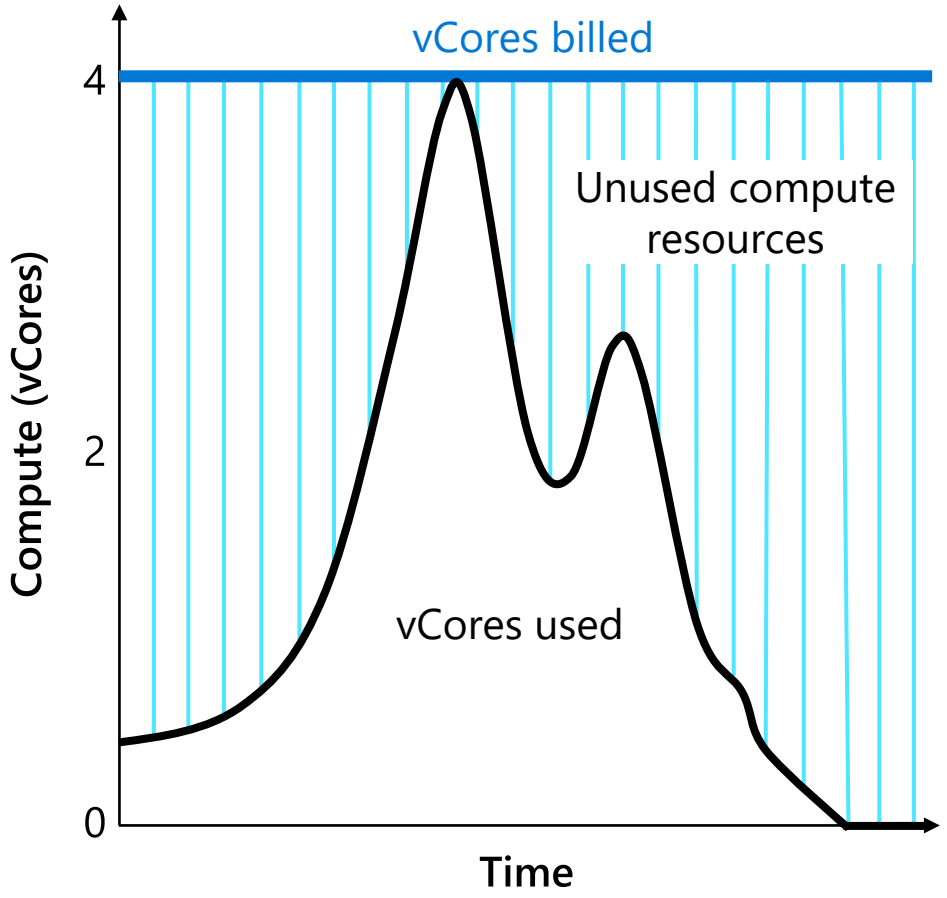
Provisioned compute databases are designed for predictable patterns and higher compute utilization

They struggle to meet high peaks in demand

They contribute to over-allocation of resources and costs during periods of inactivity or low usage

Lead to precious resources spent managing, not building

Provisioned compute with unpredictable and intermittent workloads



SQL Database serverless

On-demand flexible scale
Operate at the true rhythm of your business

Adapts compute resources to the workload without sacrificing performance

Automatically pauses and resumes

Cost-effective
Pay for performance. Period.

Pay only for compute resources you consume, on a per-second basis

Further optimize costs with configurable compute thresholds

Fully managed & intelligent
Focus on your applications, not your infrastructure

Fully-managed and intelligent database service

Built-in 99.99% availability

Best for unpredictable and intermittent workloads on single databases, such as:

Dev/test

Line of Business

E-commerce

Serverless supports common industry scenarios of sporadic or unpredictable usage

Line of business apps

Expense reporting and employee tracking apps

Procurement systems

E-commerce

Opening new marketplaces, marketing campaigns, sales promotions

Content management systems

Updating and publishing web content

Content clearinghouses that pull select content by third parties

Dev/test workloads

Handling unpredictable workload needs

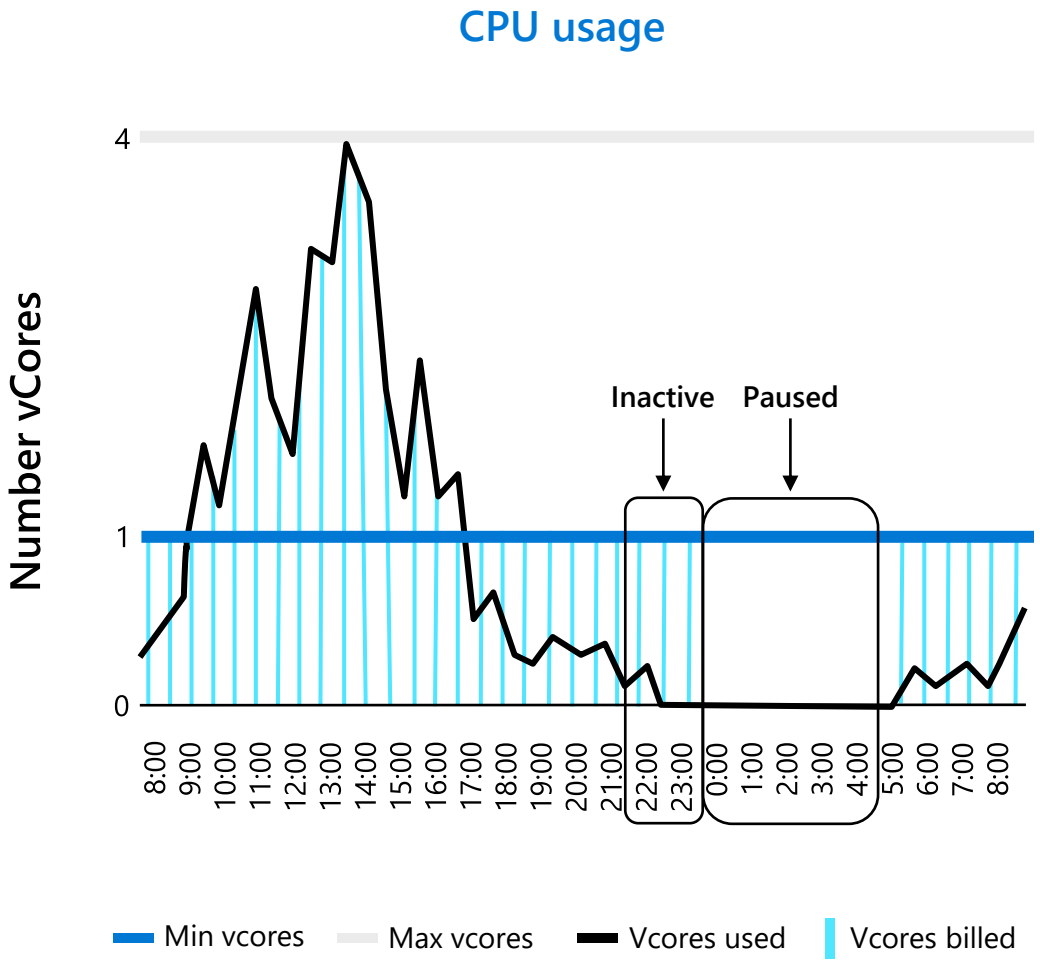
Optimize price to performance with per-second billing

Compute resources scale dynamically up or down based on workload requirements

Configure minimum and maximum vCores to define the range of available compute capacity

Use auto-pause delay to define the time period the dataset must be inactive before pausing

Pay for compute based on the vCores and memory used per second, with lowest billing based on configured vCore minimum



Provisioned compute and serverless meet different needs

Optimize compute provisioning and billing for your workload

Serverless databases...

Scale up or down to meet workload requirements, instead of pre-provisioning

Bill on a per-second basis

Common scenarios

Workloads with unpredictable and intermittent usage patterns or performance requirements

Workloads where the requirements are unknown and you can delegate compute sizing to the service

Databases with provisioned compute...

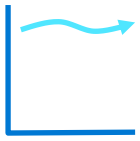
Provision compute resources upfront

Bill on an hourly basis

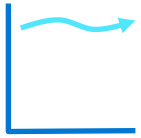
Common scenarios

Workloads with regular and substantial compute utilization

Multiple databases with bursty usage patterns that can be consolidated into a single server and use *elastic pools* for better price optimization



Choosing provisioned or serverless compute



Characteristics for provisioned compute

- More uniform resource utilization
- Need for higher compute responsiveness
- Scenarios where hourly billing granularity is ideal
- Desire to maintain resource allocation
- Interested in reserved capacity, Azure Hybrid Benefit, or elastic pools

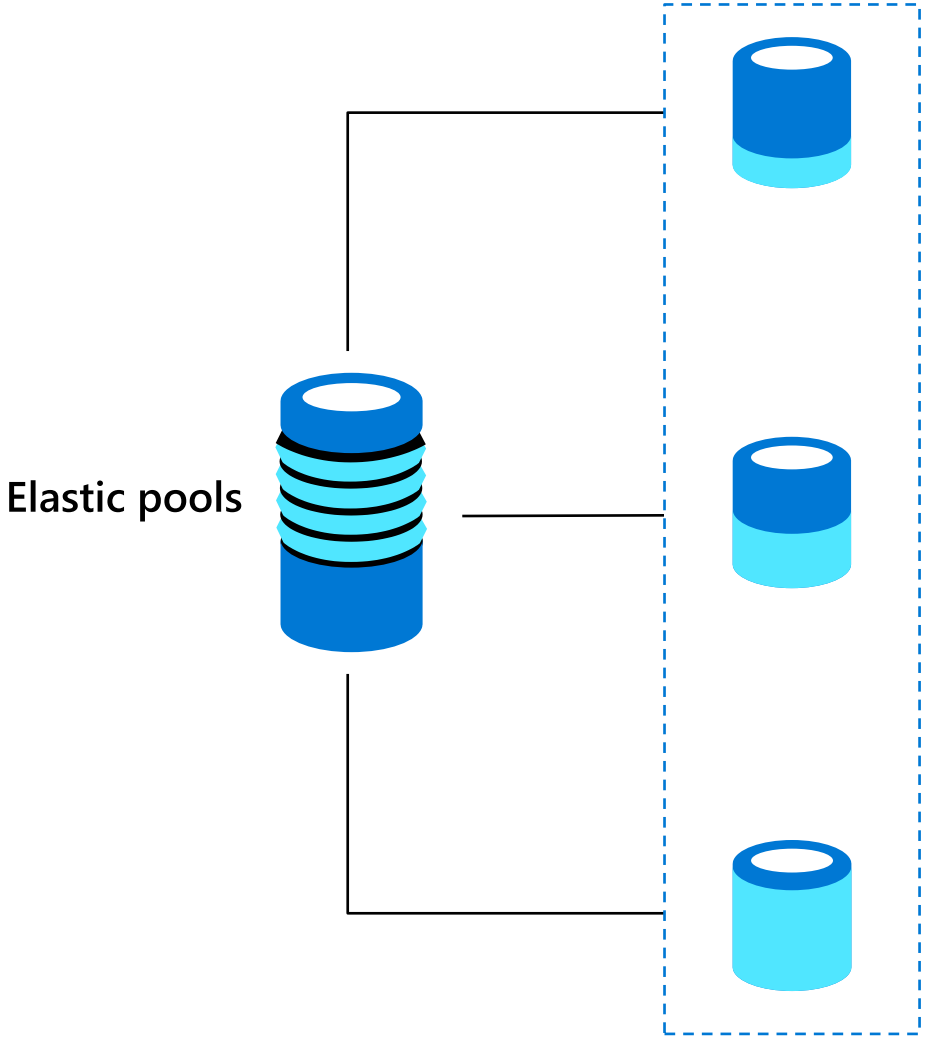
Characteristics for serverless compute

- Large shifts in usage and periods of inactivity
- Comfort with resume delay and memory reclamation
- Scenarios where per second billing granularity is ideal
- Desire to delegate resource allocation
- Currently using single databases on General Purpose service tier

Price optimization will help drive the decision between provisioned and serverless compute

When is elastic pools the right choice?

Multiple databases with unpredictable and intermittent usage patterns can be consolidated into a single server and use elastic pools for better price optimization



Resources

- Announcement blog post
- SQL Database serverless documentation
- Azure SQL Database pricing information
- Pricing calculator

Intelligent performance that learns and adapts with your workloads

Learning Objectives

Single SQL code base
In-Memory OLTP
Query Store
Extended events
Query Performance Insight
Index Advisor
Automatic tuning
Monitoring at scale
Adaptive Query Processing

Single SQL Code Base

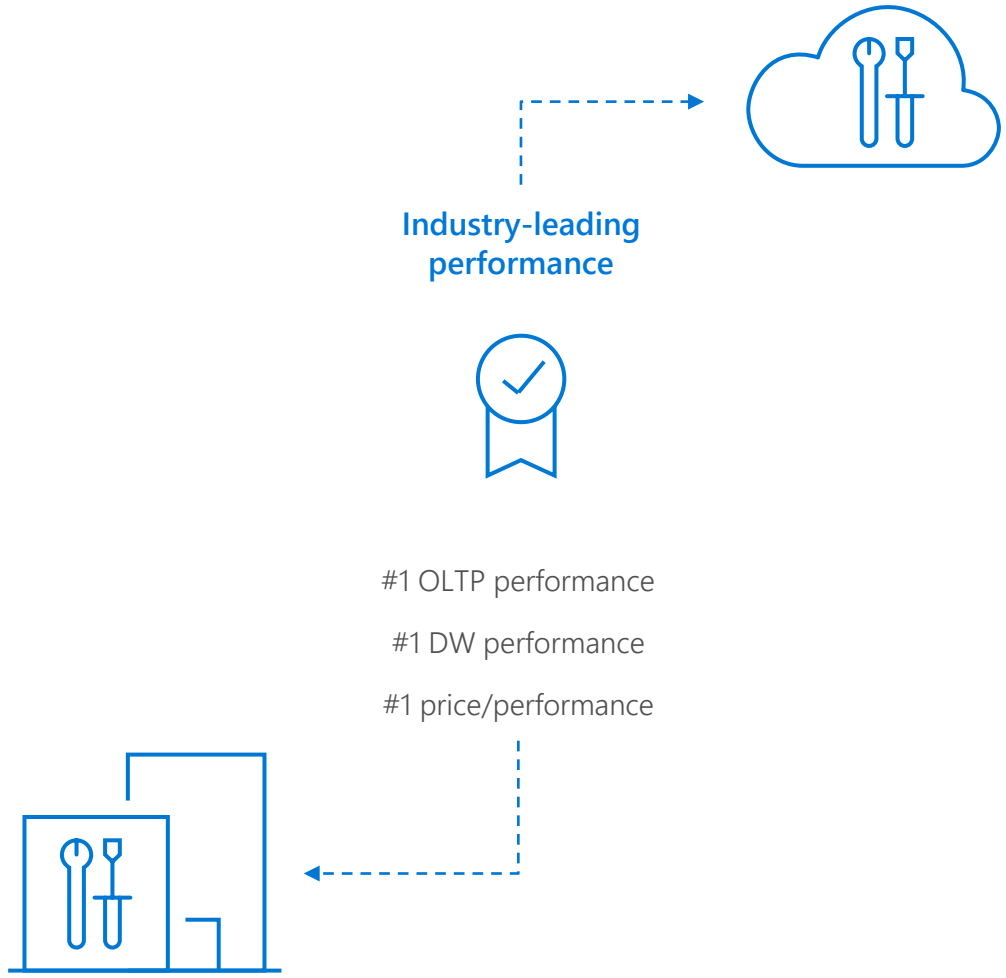
Eliminate app changes with full SQL Server programming surface

Use familiar SQL Server features in Azure SQL Database

Rapid development cycles with built-in testing across millions of databases

Innovation deployed to Azure first

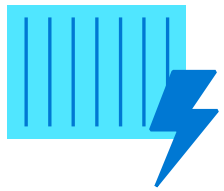
Industry-leading database engine



Breakthrough productivity and performance

Realize the benefits of real-time operational analytics

Enable scale-up with near zero downtime through cloud-born innovation



30x
faster transactions with In-Memory OLTP

100x
performance gains with in-memory analytics

Near
100% uptime with dynamic scalability

In-Memory OLTP

	High-performance data operations	Efficient business-logic processing	Frictionless scale up	Hybrid engine and integrated experience
Customer benefits				
Architectural pillars	**Main-memory optimized** • Optimized for in-memory data • Indexes (hash and range) exist only in memory • No buffer pool or B-trees • Stream-based storage	**Transact-SQL compiled to native machine code** • Transact-SQL compiled to machine code by using C code generator • Invoking a procedure is just a DLL entry-point • Aggressive compile-time optimizations	**High concurrency** • Multi-version optimistic concurrency control with full ACID support • Core engine uses lock-free algorithms • No lock manager, latches, or spinlocks	**SQL Server integration** • Same manageability, administration, and development experience • Integrated queries and transactions • Integrated high availability and backup/restore
Drivers	**Hardware trends** • Steadily declining memory price for non-volatile, random-access memory (NVRAM)	• Stalling CPU clock rate	• Processors with many cores	**Business** • Total cost of ownership

Real-time operational analytics

Capabilities

In-memory columnar index over in-memory/disk-based OLTP tables

Enhanced OLTP Transact-SQL surface area

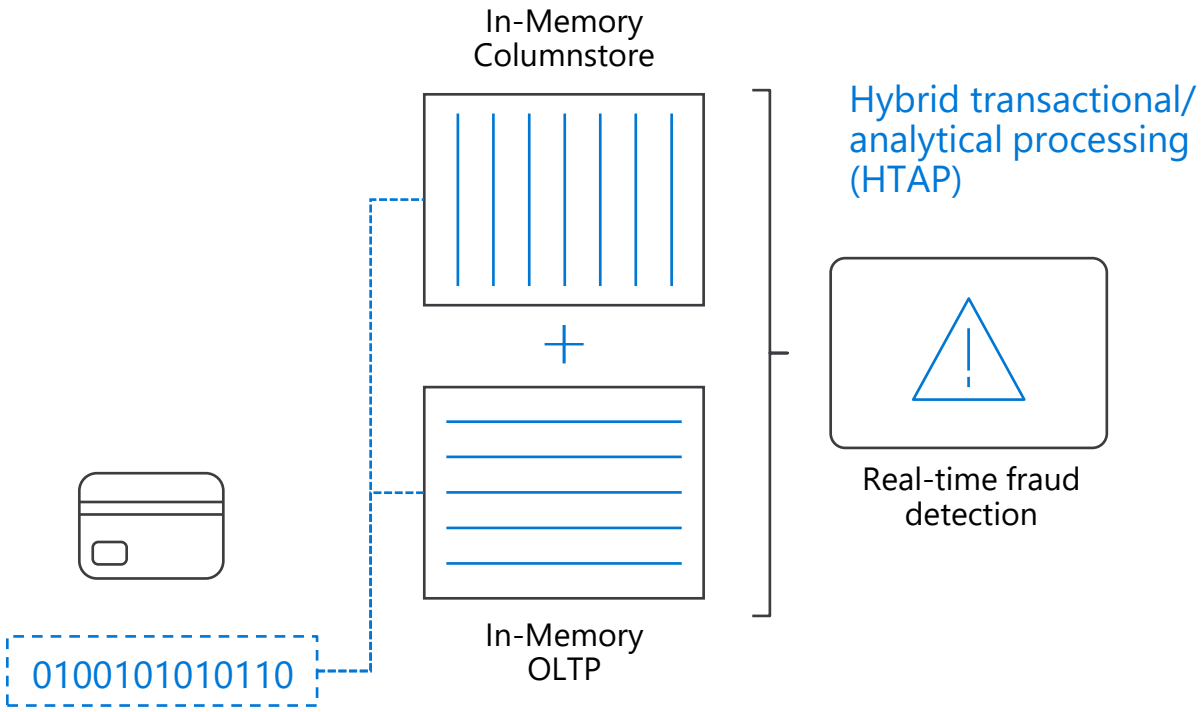
Scale to higher compute and memory

Benefits

Eliminate the need for ETL and a data warehouse

30x faster transactions and 100x better query performance

Run analytics in real time on up-to-date data



Query Store for comprehensive performance

Flight data recorder for your database

Captures history of queries, plans, and runtime statistics

Quickly finds performance differences caused by query plan changes

Separates data by time window

Query Store use

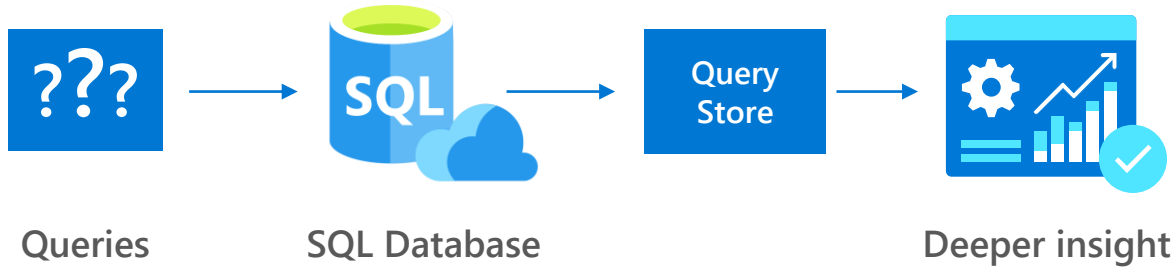
Enabled by default

Find regressed queries

Identify top resource-consuming queries

Optimize ad-hoc workloads

Streamline application upgrades



Working with Query Store

DB-level feature exposed through T-SQL extensions

ALTER DATABASE

Catalog views (settings, compile, and runtime stats)

Stored procs (plan forcing, query/plan/stats cleanup)

```
/* (1) Turn ON Query Store */

ALTER DATABASE MyDB SET QUERY_STORE = ON;

/* (2) Review current Query Store parameters */
SELECT * FROM sys.database_query_store_options

/* (3) Set new parameter values */
ALTER DATABASE MyDB
SET QUERY_STORE (
  OPERATION_MODE = READ_WRITE,
  CLEANUP_POLICY = (
    STALE_QUERY_THRESHOLD_DAYS = 30
  ),
  DATA_FLUSH_INTERVAL_SECONDS = 3000,
  MAX_STORAGE_SIZE_MB = 500,
  INTERVAL_LENGTH_MINUTES = 15
);

/* (4) Clear all Query Store data */
ALTER DATABASE MyDB SET QUERY_STORE CLEAR;

/* (5) Turn OFF Query Store */
ALTER DATABASE MyDB SET QUERY_STORE = OFF;
```

Increasing value of captured data

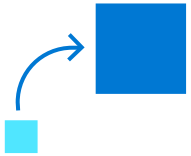
Configuration options

Data compression

Storing plans in a compressed format

Time-based cleanup

Removing stale queries automatically



Size-based cleanup

Cleaning up least relevant data before exceeding quota

Query Capture Mode

Capturing only relevant queries

ALL: capture all queries

NONE: stop capturing new queries

AUTO (magic knob): collect only highly relevant queries

Stats captured by Query Store

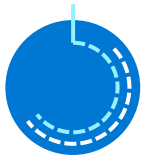
Compile-time stats	Query text
	Semantic-affecting settings
	Containing objects: SP, TVF, trigger
	Parametrization type
	Compilation, binding, and optimization stats
	Query plan plus initial and last compile/execute times
Run-time stats (aggregated on an interval)	Count of executions and first/last execution time
	AVG, LAST, MIN, MAX, and STDEV for **{metrics}** • Duration • CPU time • Logical I/O reads and writes • Physical I/O reads • DOP • Memory grants • Number of rows

Extended Events

Enhanced monitoring and troubleshooting for Azure SQL Database

Performance issues
SQL statement executions
Full-text related errors

Targets that can capture results from your event sessions



Ring buffer target

Briefly holds event data in memory

Event counter target

Counts all events that occur during an extended events session

Event file target

Writes complete buffers to an Azure Storage container

Analyze resources with Query Performance Insight

Review top CPU consuming queries

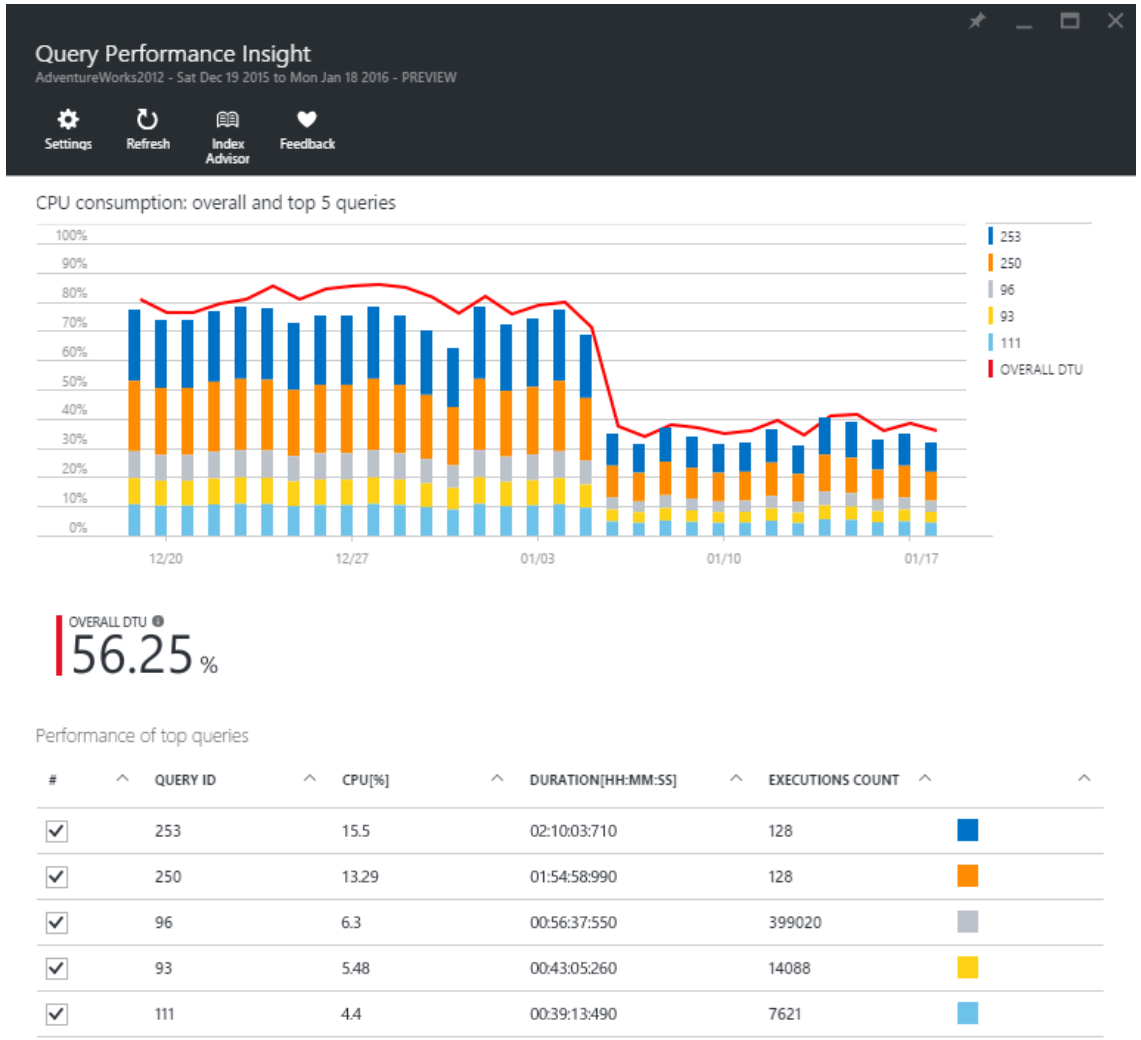
Customize your view by selecting observation interval, number of queries, and aggregation type

View aggregated statistics about your workload including total duration and number of executions

View individual query details

Get granular view on query execution intervals

View query text



Understand and tune your database with Query Performance Insight

Insights into resource consumption

Query Performance Insights shows resource consumption of top resource-consuming queries over time, and helps pinpoint the potential issues with additional details for each query

Hints for optimization

Query Performance Insights gives tuning hints for top resource consumers

Query Performance Insights can help users with the following:

Answer the question “Where are my resources spent?” and understand the impact of top queries on the resource consumption of the database over time

Identify the queries to fix, before (or after) they become a problem

Azure SQL Database Advisor improves performance

Index tuning recommendations tailored to each DB

Recommendations are based on the observed usage, and evolve as the DB workload changes

Support for CREATE and DROP index, more types of recommendations underway

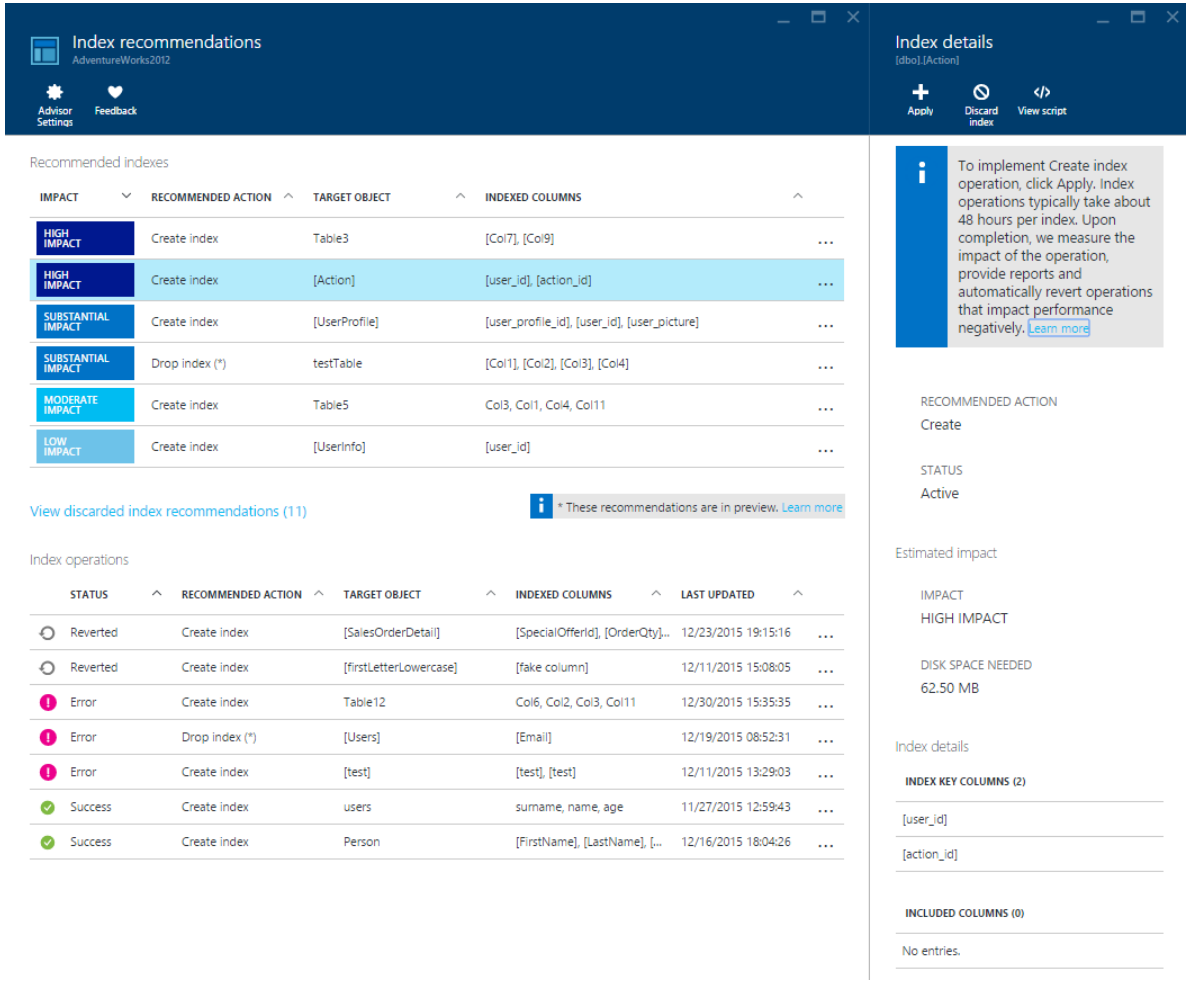
Drop index recommendations

Recommendations are based on the observed usage of index not used

Intelligent service for implementing and validating index recommendations

Full-auto mode takes care of indexes for your DB

Manual “review and apply” mode for full control



Continuously optimized by the platform | Automatic tuning

One-click to enable

Prevent and mitigate issues

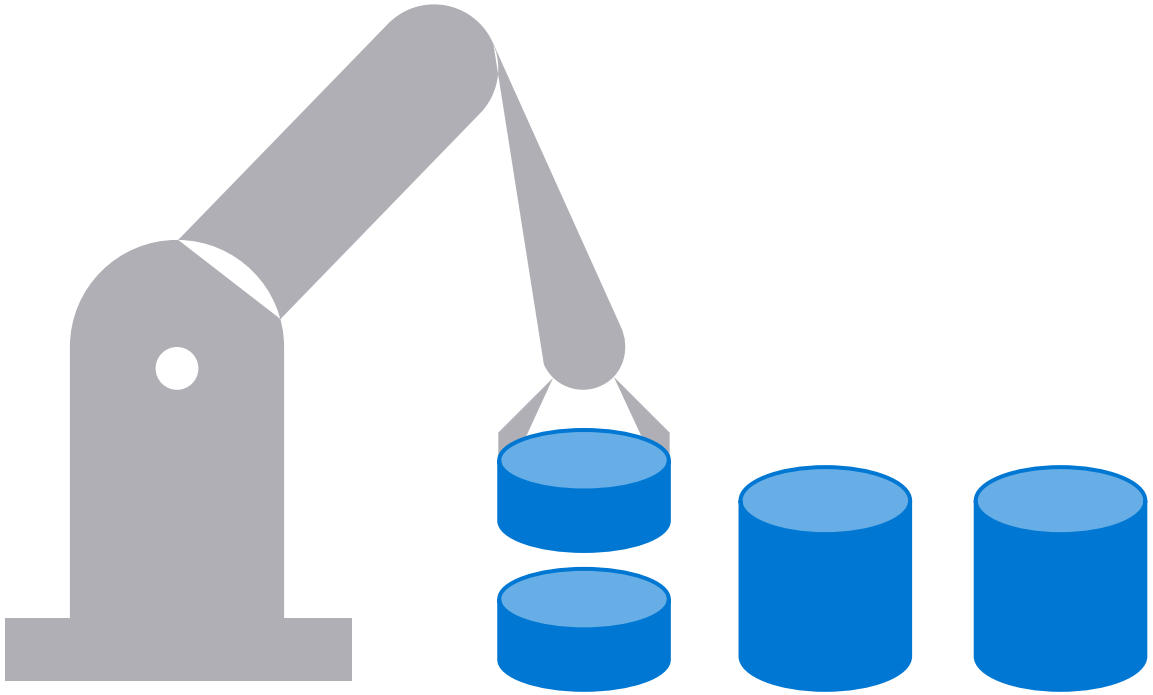
No app changes needed

Tuning actions

Create missing indexes

Drop unused/duplicate indexes

Force last good plan



Azure SQL Analytics

Monitoring Azure SQL Database at scale

Integrated with Operations Management Suite

Monitor entire Azure SQL Database estate across multiple subscriptions

Provides a view into raw telemetry

Monitored activities

Resource usage

Query performance

SQL errors

Timeouts

Blocking

Intelligent insights

Intelligent insights

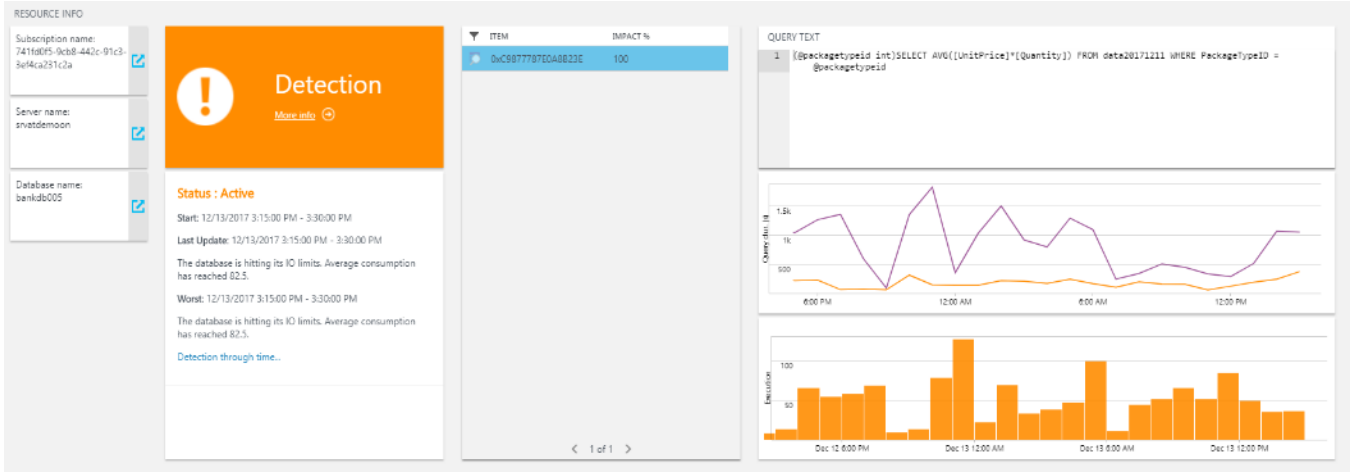
Insights into databases performance

Built-in intelligence that continuously monitors database usage

Uses Artificial Intelligence to detect disruptive events that cause poor performance

Generates diagnostic log with intelligent assessment for disruptive events

Performs root cause analysis and provides remediation where possible for performance improvements



Adaptive Query Processing

Interleaved execution

Materialize estimates for multi-statement table valued functions (MSTVFs)

Downstream operations will benefit from the corrected MSTVF cardinality estimate

Batch mode memory grant feedback

Adjust memory grants based on execution feedback

Remove spills and improve concurrency for repeating queries

Batch mode adaptive join

Defer the choice of hash join or nested loop until after the first join input has been scanned

Uses Nested Loop for small inputs, Hash Joins for large inputs

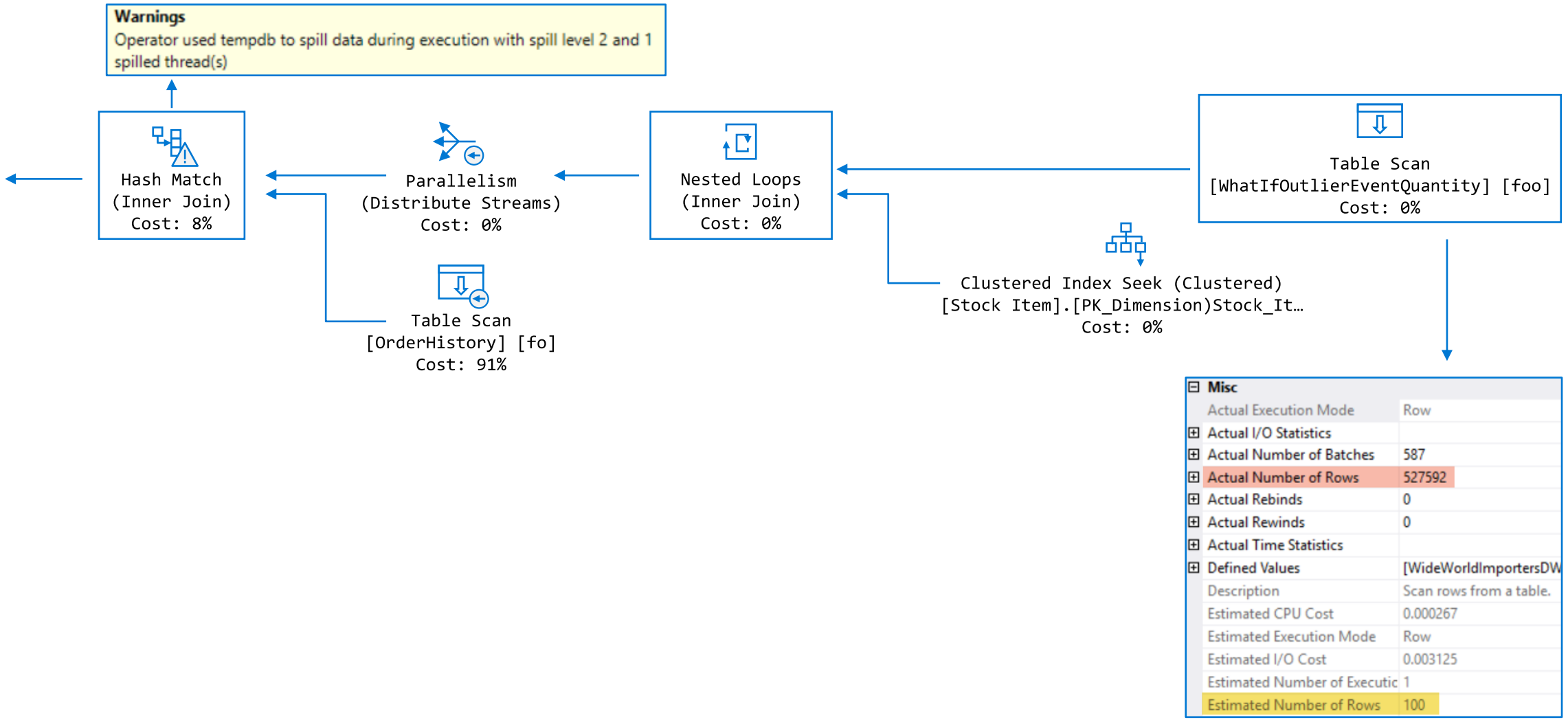
Challenges associated with mis-estimation

- Slow query response time due to bad plans
- Excessive resource utilization (CPU, Memory, IO)
- Reduced throughput and concurrency
- T-SQL refactoring for off-model statements

Cardinality estimation and plan quality

```
SELECT  [fo].[Order Key], [fo].[Description], [fo].[Package],
        [fo].[Quantity], [foo].[OutlierEventQuantity]
FROM    [Fact].[OrderHistory] AS [fo]
INNER JOIN [Fact].[WhatIfOutlierEventQuantity]('Mild Recession',
                        '1-01-2013',
                        '10-15-2014') AS [foo] ON [fo].[Order Key] = [foo].[Order Key]
                       AND [fo].[City Key] = [foo].[City Key]
                       AND [fo].[Customer Key] = [foo].[Customer Key]
                       AND [fo].[Stock Item Key] = [foo].[Stock Item Key]
                       AND [fo].[Order Date Key] = [foo].[Order Date Key]
                       AND [fo].[Picked Date Key] = [foo].[Picked Date Key]
                       AND [fo].[Salesperson Key] = [foo].[Salesperson Key]
                       AND [fo].[Picker Key] = [foo].[Picker Key]
INNER JOIN [Dimension].[Stock Item] AS [si]
    ON [fo].[Stock Item Key] = [si].[Stock Item Key]
WHERE   [si].[Lead Time Days] > 0
        AND [fo].[Quantity] > 50;
```

Cardinality estimation and plan quality



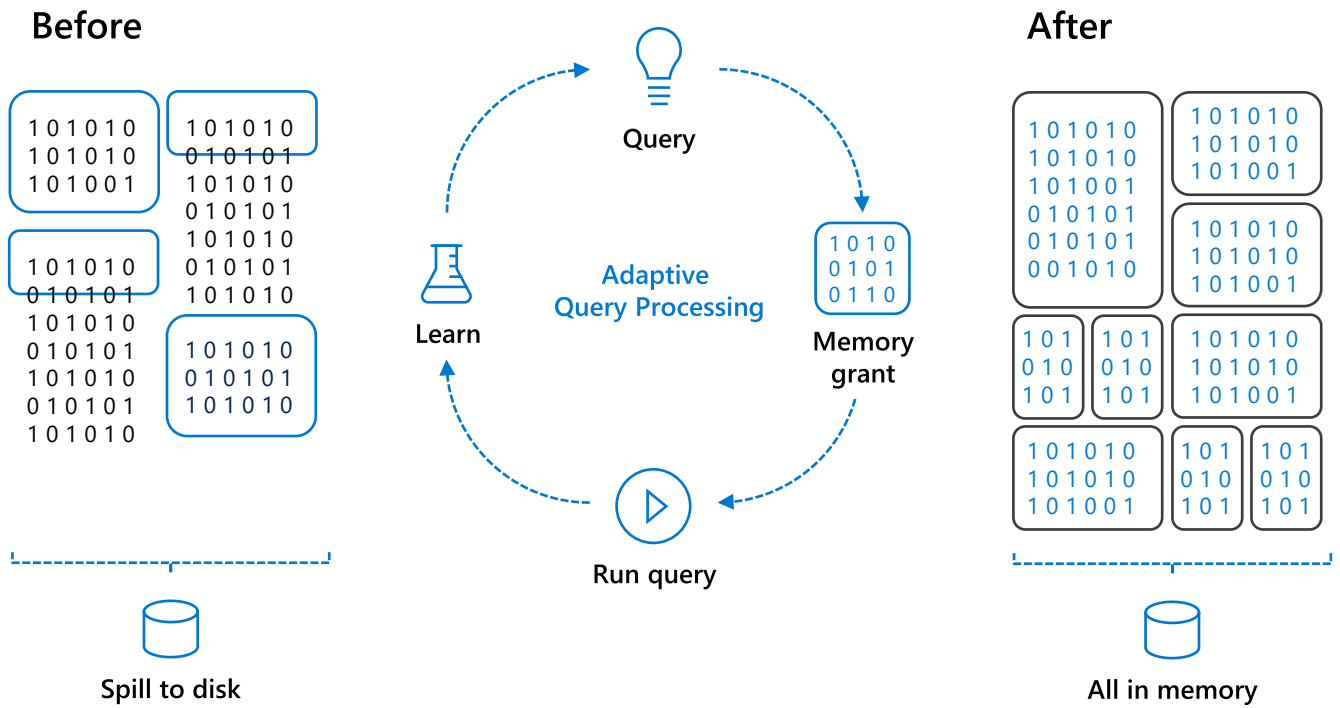
Optimized query processing

Improved efficiency with Adaptive Query Processing

Optimize memory grants for repeatable queries to avoid over or under allocating

Adjust data join strategy for small or large tables to speed joins

Batch mode for memory grants feedback and adaptive joins



Scale on the fly with no application downtime

Learning Objectives

Scalable performance

Predictable performance

Pay for what you need Real-time analytics

Steps to optimize your service tier

Scalable performance with LITTLE to no downtime

Predictable performance
Guaranteed resources allow for a predictable performance experience

In-Memory OLTP
Up to 30x performance gains with In-Memory OLTP

Pay for what you need, when you need it
Easily scale up app resources to accommodate growth periods or peak workload demand

Real-time analytics
Get real-time insights and 100x performance gains with real-time operational analytics

Service tier selection overview

Choosing the right service tier/performance level is dependent on:

Feature usage and features available in different service tiers

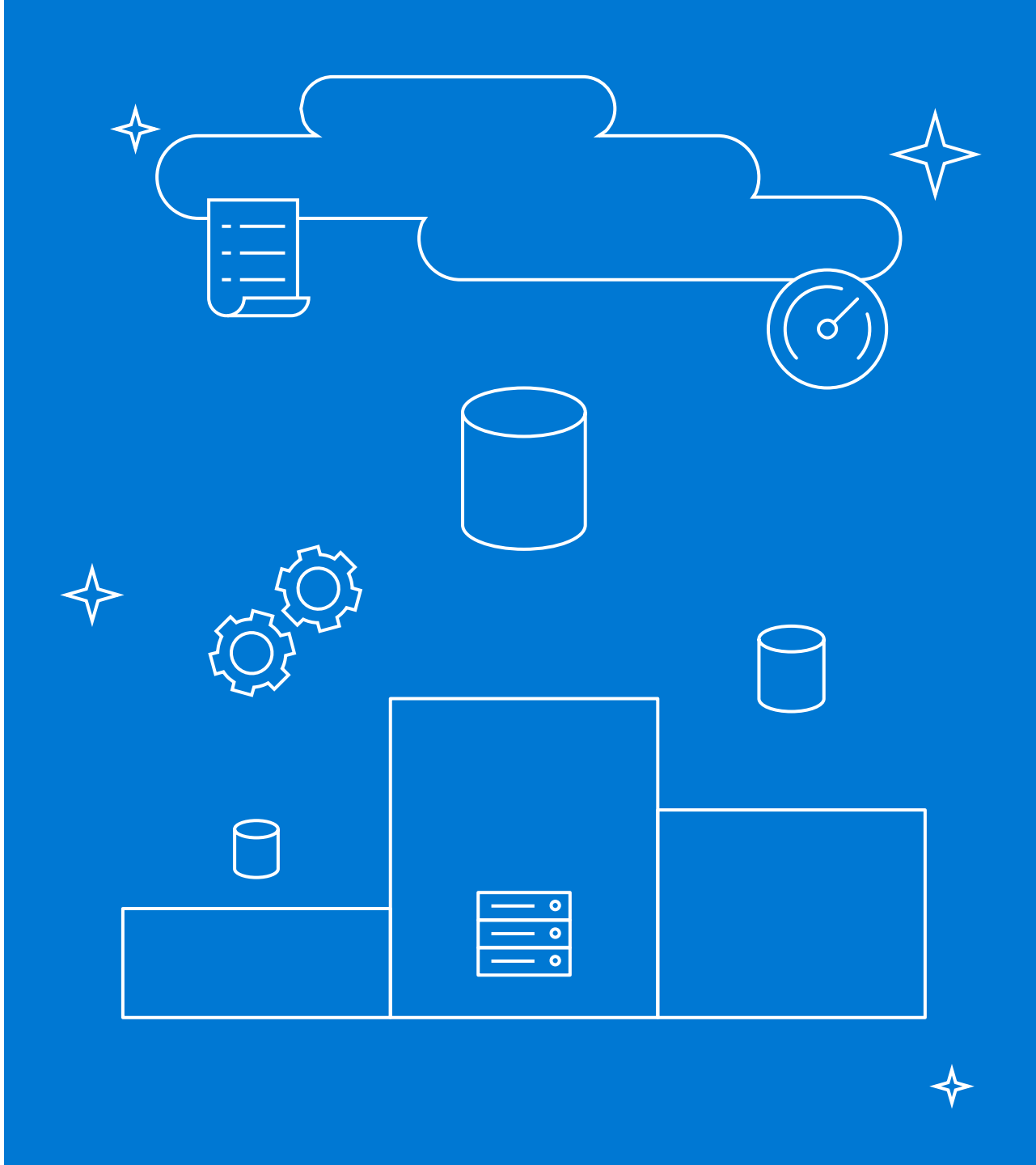
Resource usage and bounding boxes of different performance levels

Selection should take into account:

Minimum database features required

Resource consumption

Database size

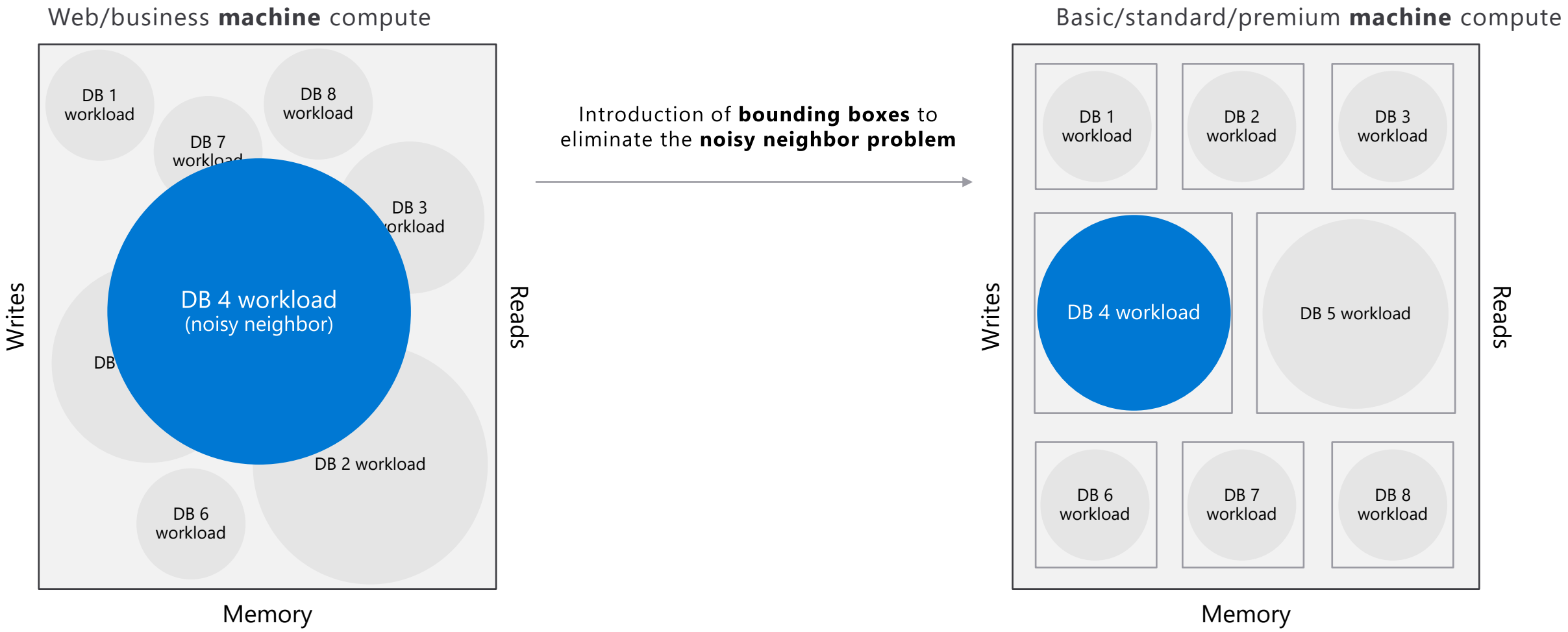


Upgrade to new service tier/performance level

After you determine the appropriate service tier and performance level for your web or business database, you can choose from multiple ways to upgrade your database to the new tier.

Azure Management Portal	On your database dashboard page, click the Pricing tier tab
Azure PowerShell	Use the Set-AzureSqlDatabase cmdlet
Service Management REST API	Use the Update Database command
Transact-SQL	Use the ALTER DATABASE (Transact-SQL) statement

Improved performance predictability



Managing large numbers of databases

Predictable workloads

Single databases or partitioned data across multiple databases

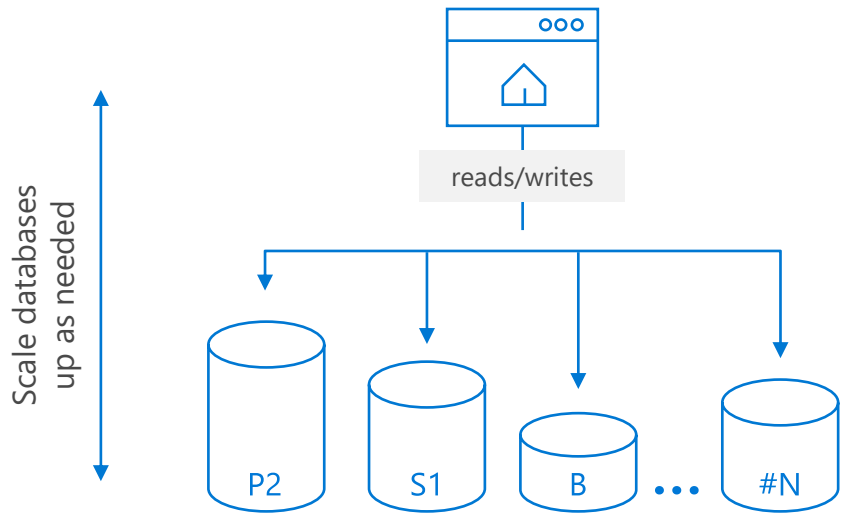
Scale between service tiers and performance levels as capacity needs fluctuate

Unpredictable workloads

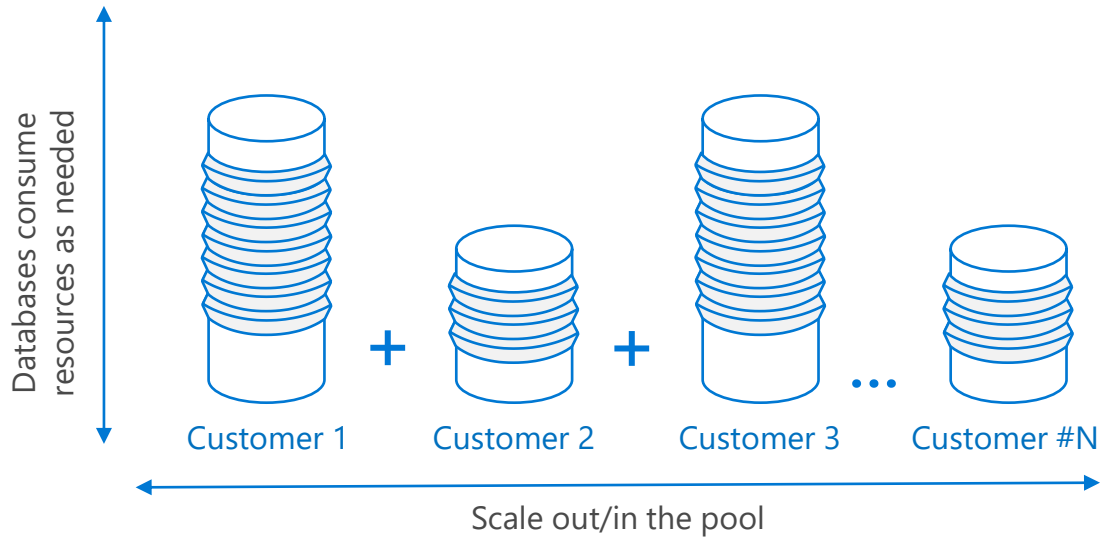
For large numbers of databases with unpredictable performance demands

Pool resources are shared between these databases

Single database or partitioned databases



Elastic Database Pool



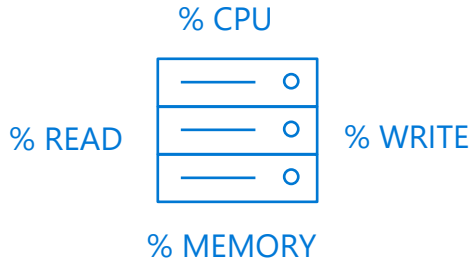
Designed for predictable performance

Redefined

Across the service tiers, each performance level is assigned a defined level of throughput for a streamlined experience

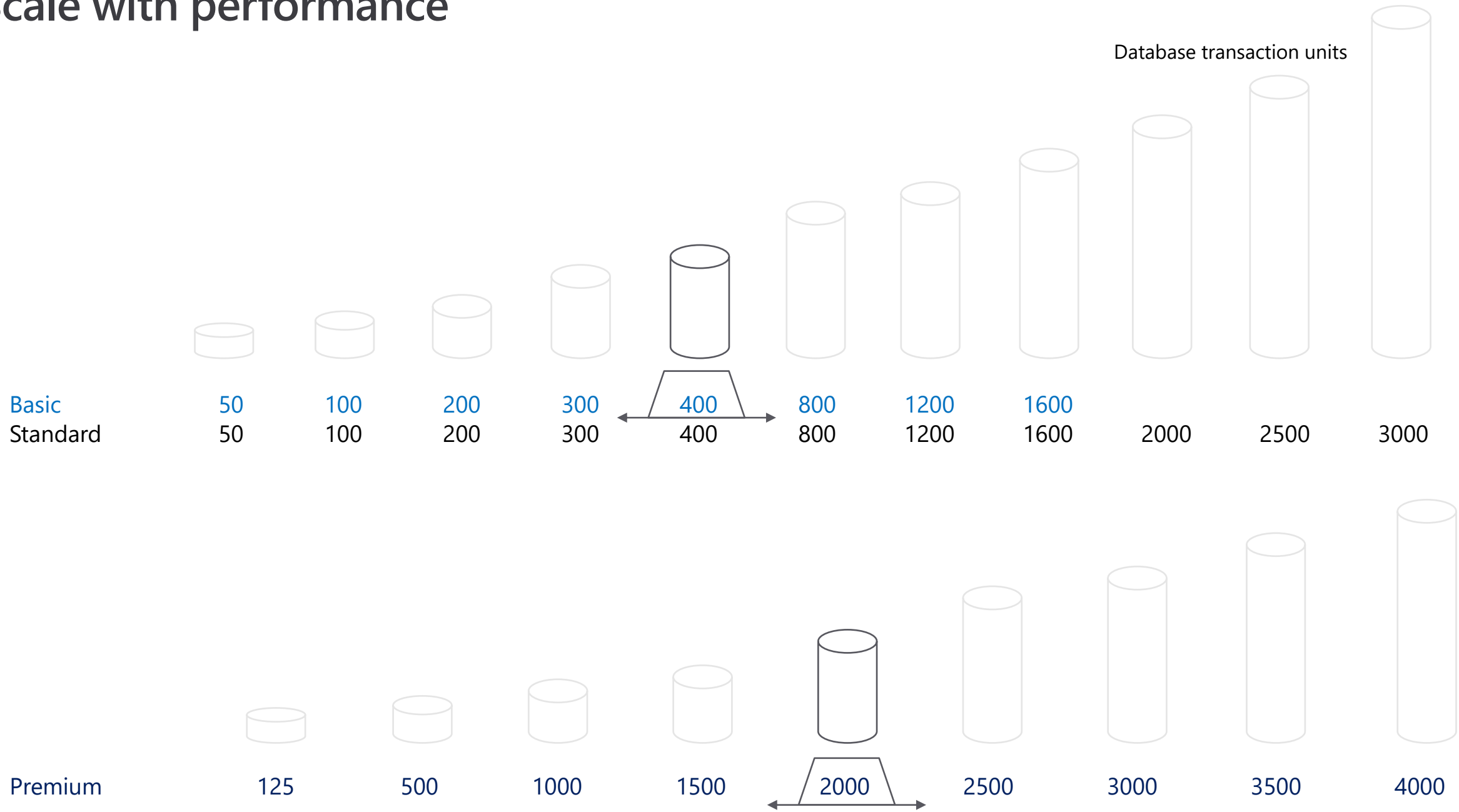
Measure of power

Introducing the Database Transaction Unit (DTU), which represents the relative power of databases based on a real-world measure: **the database transaction**



DTU represents a set of operations that are typical for an online transaction processing (OLTP) request, and then measured by how many transactions could be completed per second under fully loaded conditions

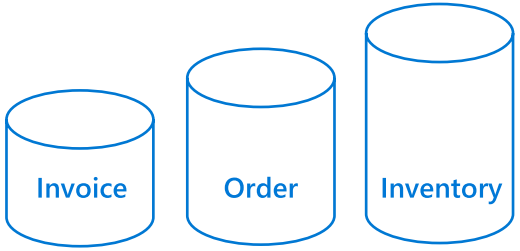
Scale with performance



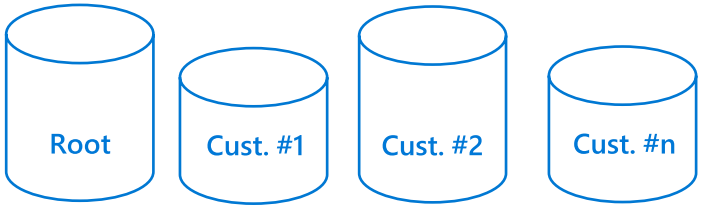
Common database scalability patterns

Single large database

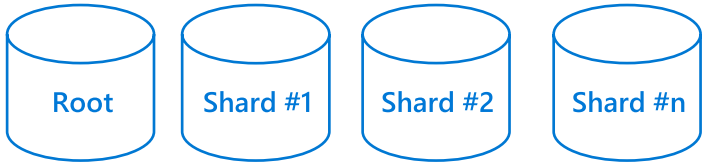
Vertically partitioned



1 tenant: 1 database (SaaS ISV)



Other partitioning scheme



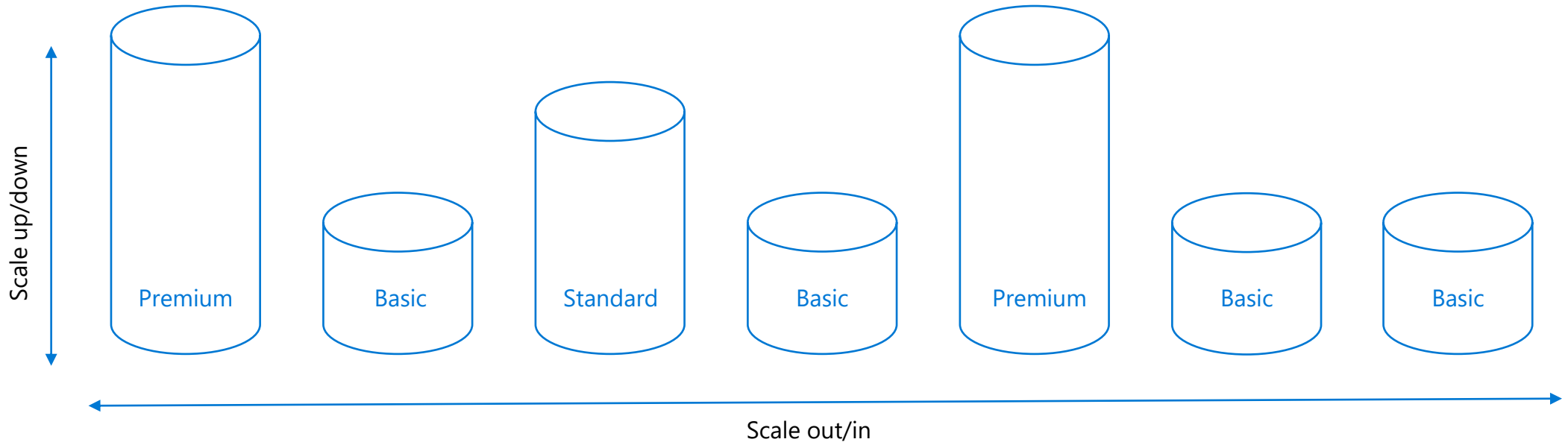
Scalability options in Azure SQL Database

Vertical: scale up or scale down

Change service tiers for a given database as capacity needs fluctuate

Horizontal: scale out or scale in

Add or remove databases (sharded and/or in a pool) as more or less capacity is needed



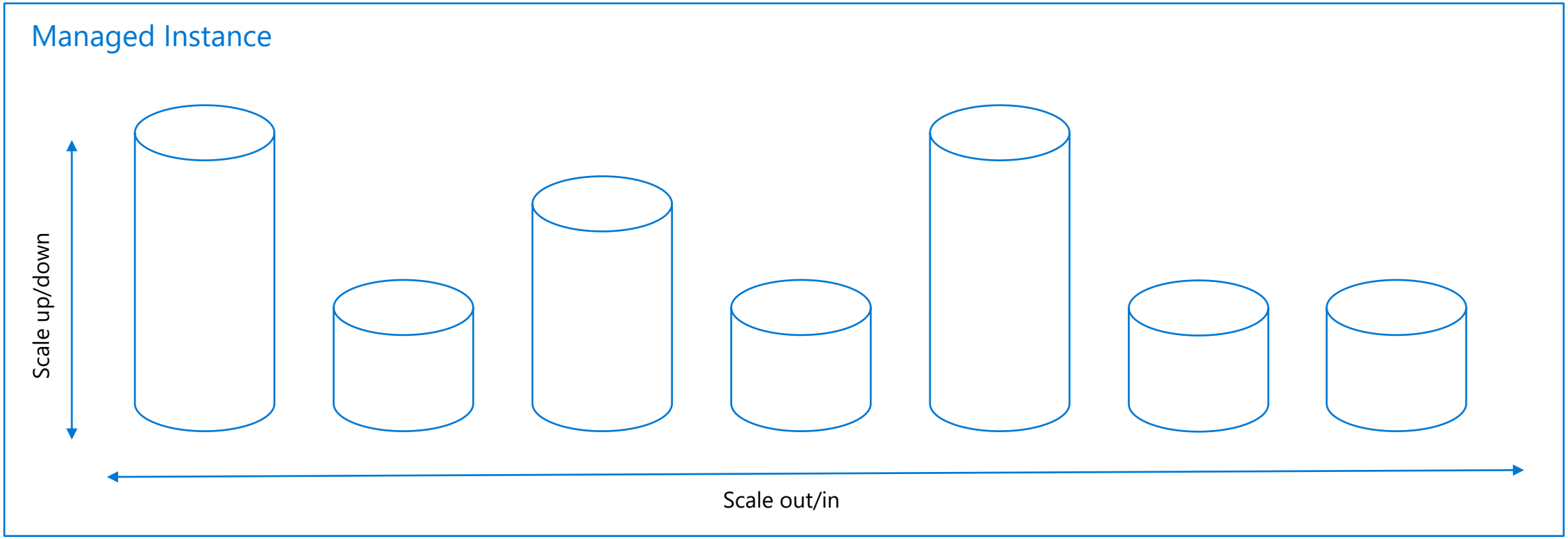
Scalability options for Azure SQL Database Managed Instance

Vertical: scale up or scale down

Change service tiers to provide a larger resource pool for the entire managed instance

Horizontal: scale out or scale in

Add or remove databases and manage performance using resource governor



Resource monitoring

master.sys.resource_stats

Based on 5 minute averages

userdb.sys.dm_db_resource_stats

Based on 15 second averages

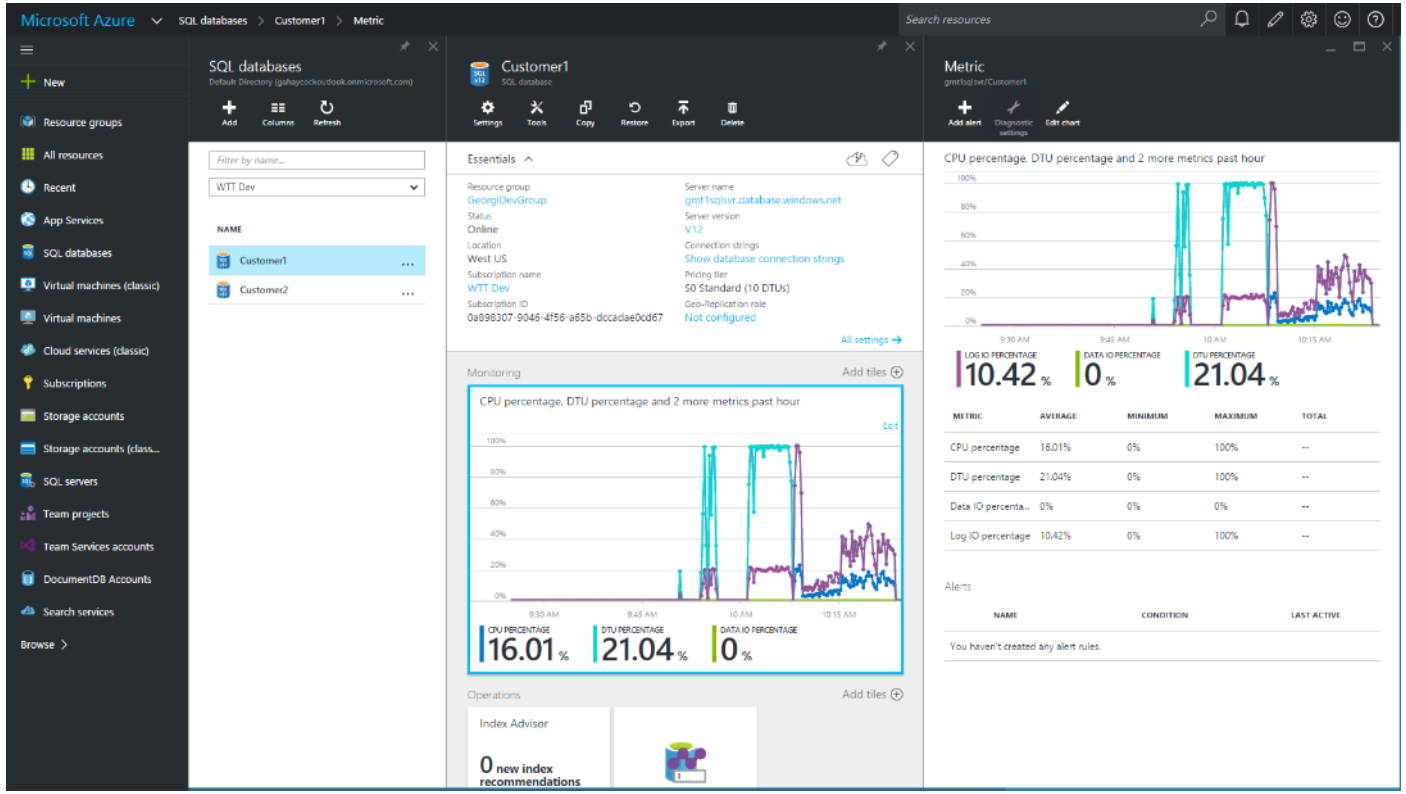
Percentages relative to performance level

Accessible though Azure Portal

Enables configuration of alerting

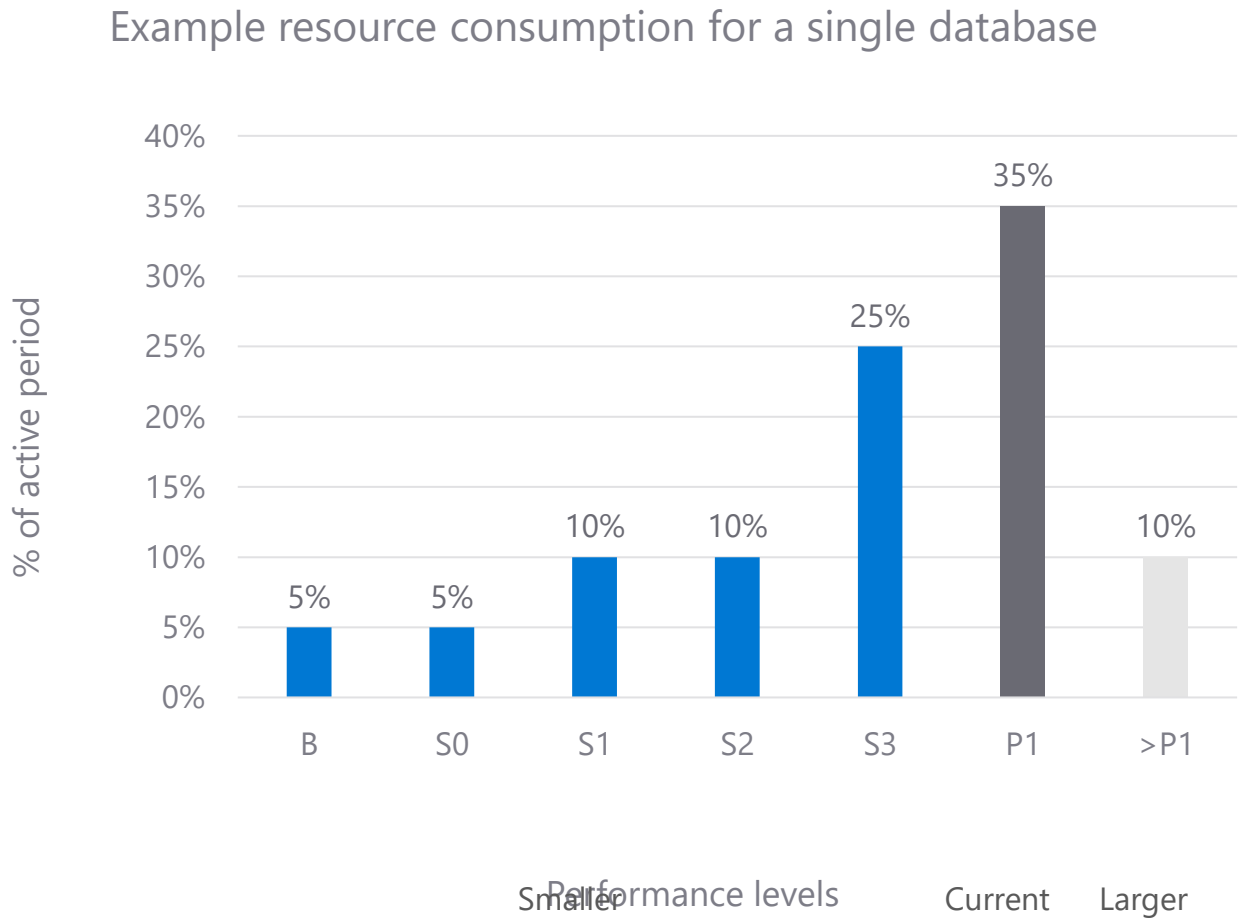
Intelligent Insights

Monitor performance across your entire Azure SQL Database estate



Resource consumption analysis

Resource consumption distribution shows what percentage of an active period the database was consuming resources within the boundaries of different performance levels.



Continuous data collection

Large data ingestion workloads

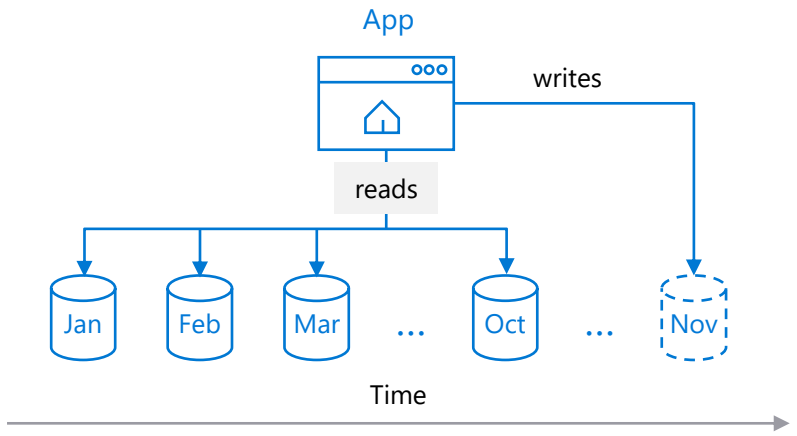
Mostly append only (Cola bottling analogy)

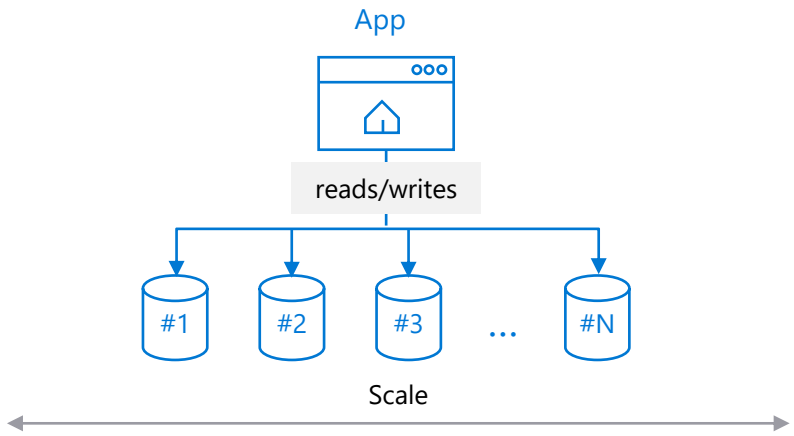
Two options, depending on application query patterns

Querying ranges of last X days of data (for example, trend analysis)

Randomly extracting data from the entire database (for example, look up a certain January event)

Typically requires fan-out queries



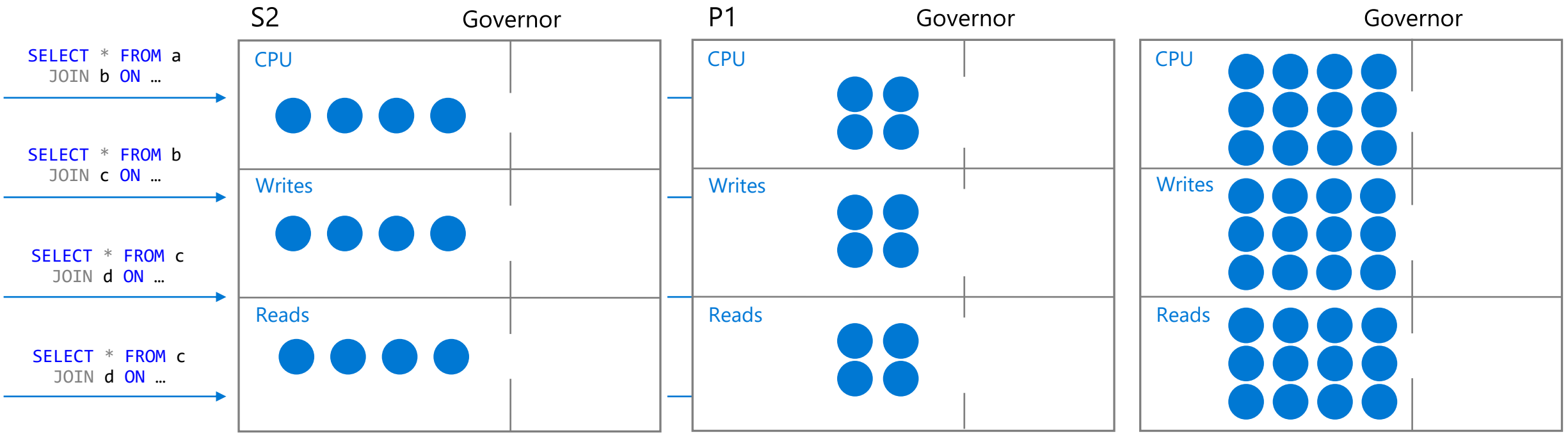


Resource governance

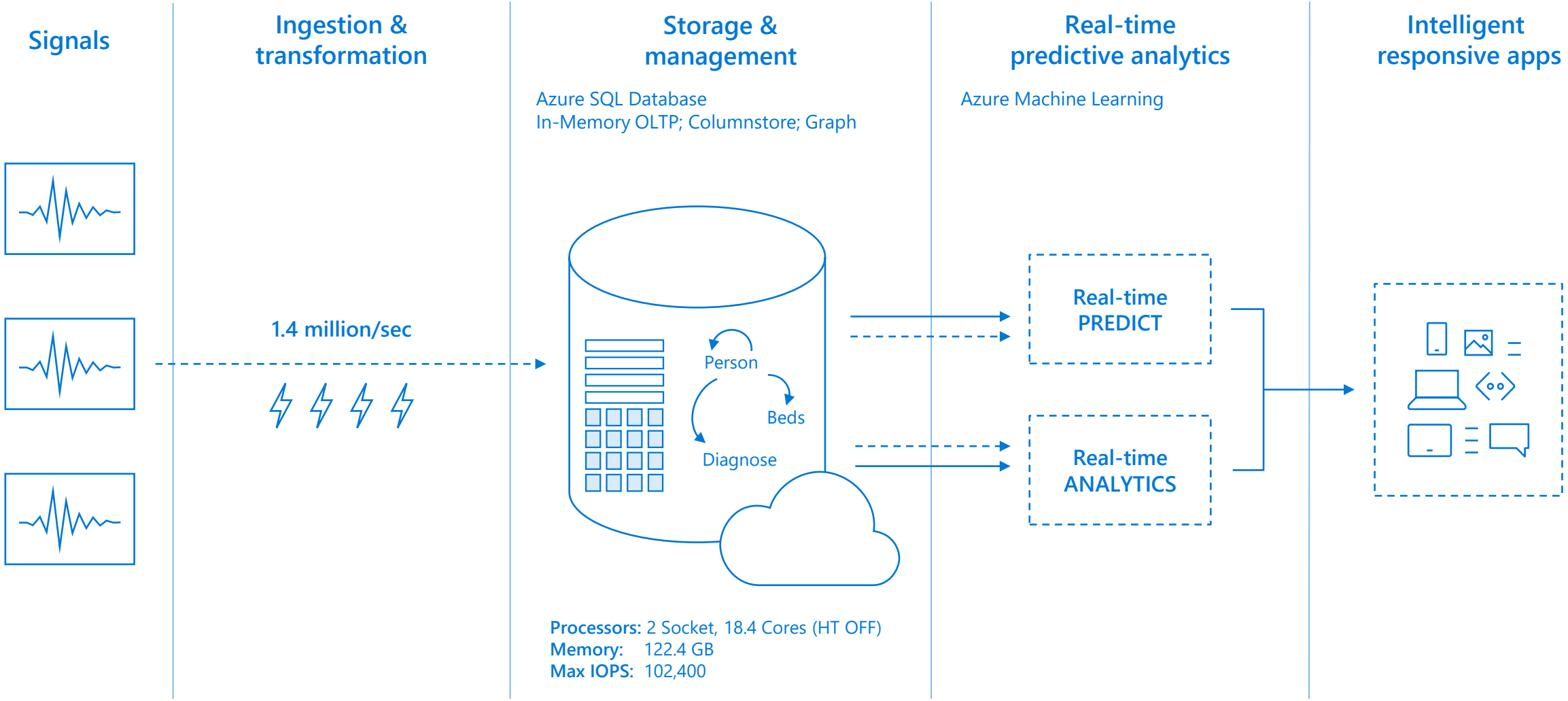
Simulate the behavior of a dedicated small computer

Resource requests are queued instead of rejected

Overloading can result in long-running transactions and command timeouts



No-fuss Performance & Intelligence



Ensure Business Continuity for your mission critical workloads

Learning Objectives

Point-in-Time Restore

Geo-Replication

Azure Data Sync

Built-in High Availability

Zone Redundancy

Software as a service

Elastic database models

SaaS Patterns

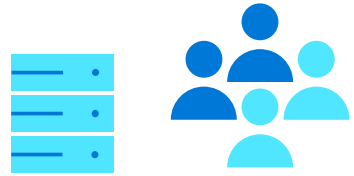
Roles and responsibilities

Azure SQL Database

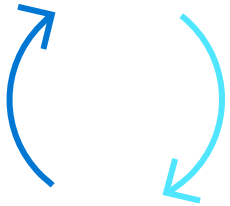
Geo-distributed service
Customer metadata protection and recovery
Transparent high availability and data protection from local platform failures
Automatic geo-distributed backups
Automatic data synchronization of geo-replicated databases
Platform compliance testing and certification
Alert to impacted customers about server degradation during regional failures

Customer (subscription owner)

Detecting user errors and initiating point-in-time restore
Planning, database prioritization, and region selection for disaster recovery
Initiating geo-restore to selected region
Initiating failover of geo-replicated databases
Application disaster recovery drills



Microsoft



You

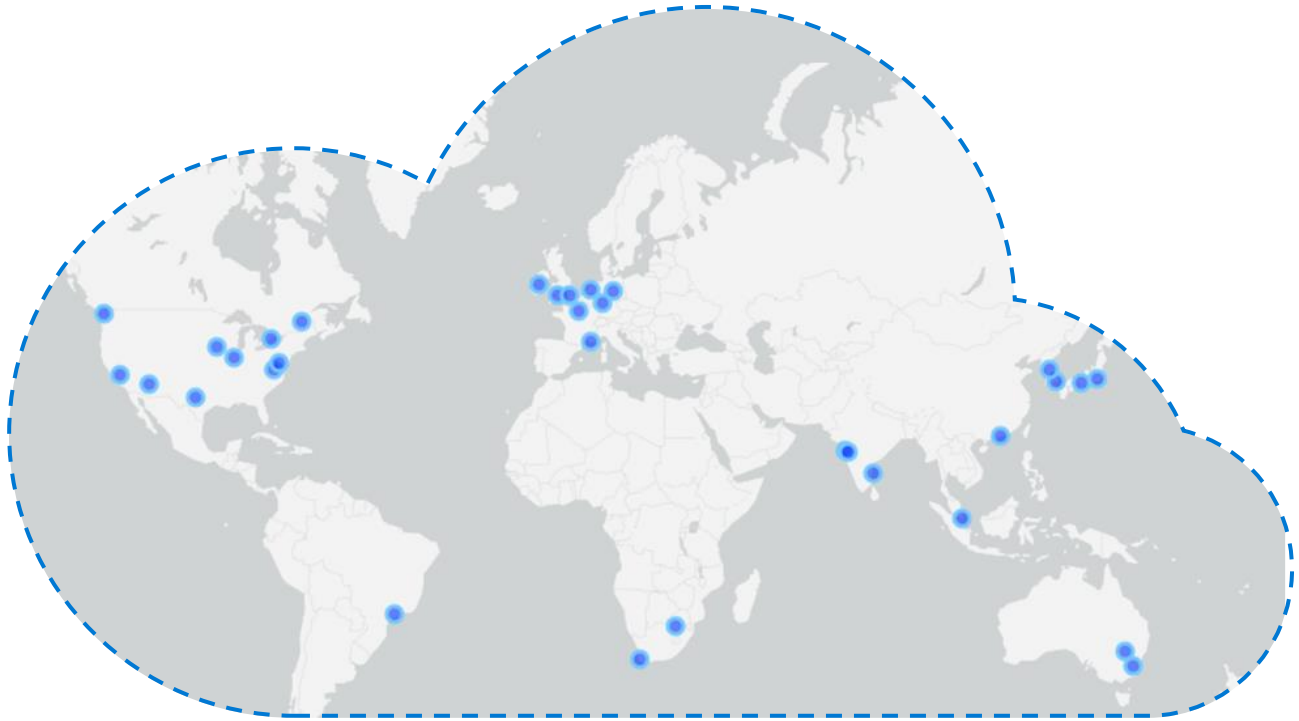
Microsoft-backed for your peace of mind

Peace of mind over your cloud investments

Built-in regional database replicas for additional protection

Uptime SLA of 99.995%*

Single support vendor across Azure cloud services



*Web & Business tiers remain backed by 99.9% uptime SLA.

Long-term data retention

Automatically created with LTR capability in Azure SQL Database

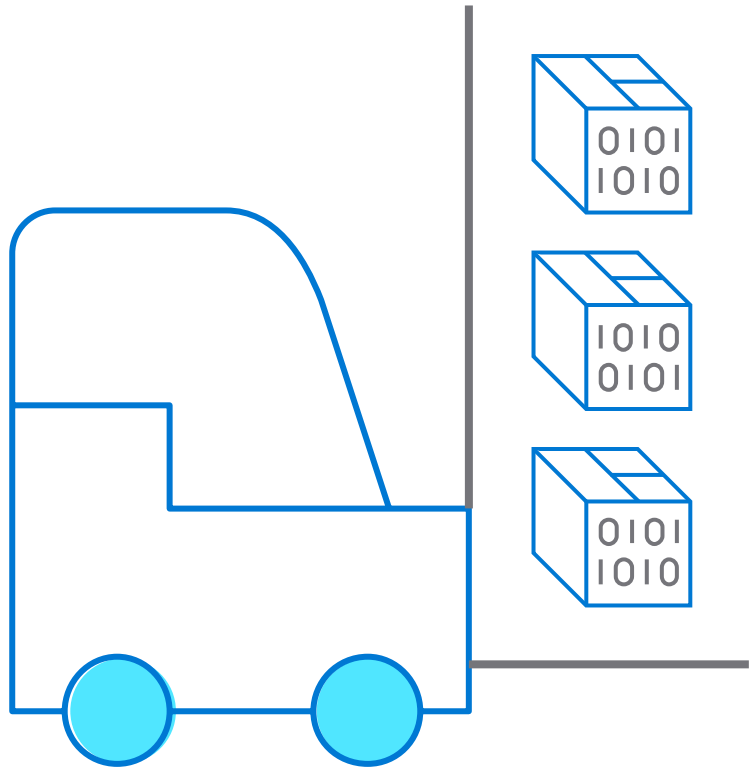
Full database backups

Store backups for up to for up to 10 Years

Read-access geo-redundant storage (RA-GRS)

Export a database

Generate a BACPAC in external storage and hydrate as needed



Point-in-time restore

Automatic backups

Full backups weekly, differential backup daily, log backups every 5 minutes

Daily and weekly backups automatically uploaded to geo-redundant Azure Storage

Self-service restore

Point-in-time up to a second granularity

REST API, PowerShell, or Azure portal

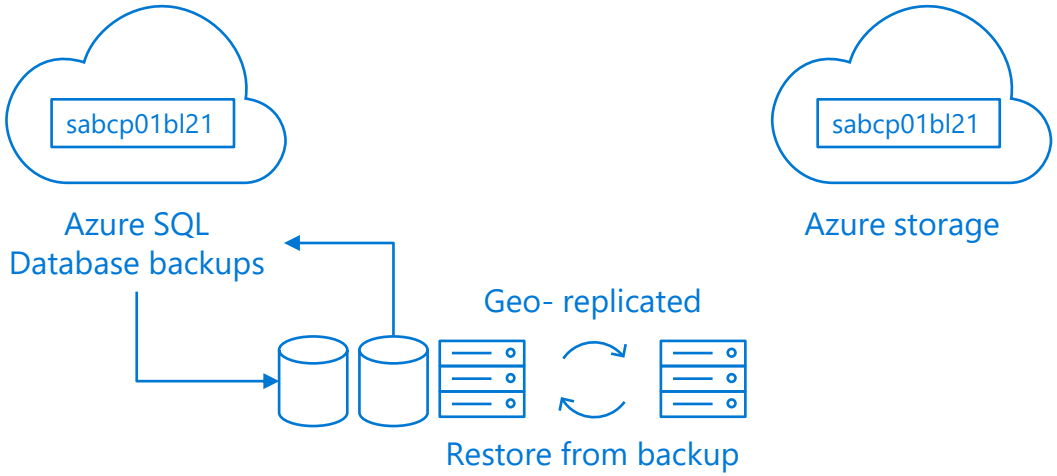
Creates a new database in the same logical server

User-controlled retention policy

7 days default retention in all service tiers

Up to 35 days of additional retention if required at additional cost

Choice of storage tier for data sovereignty



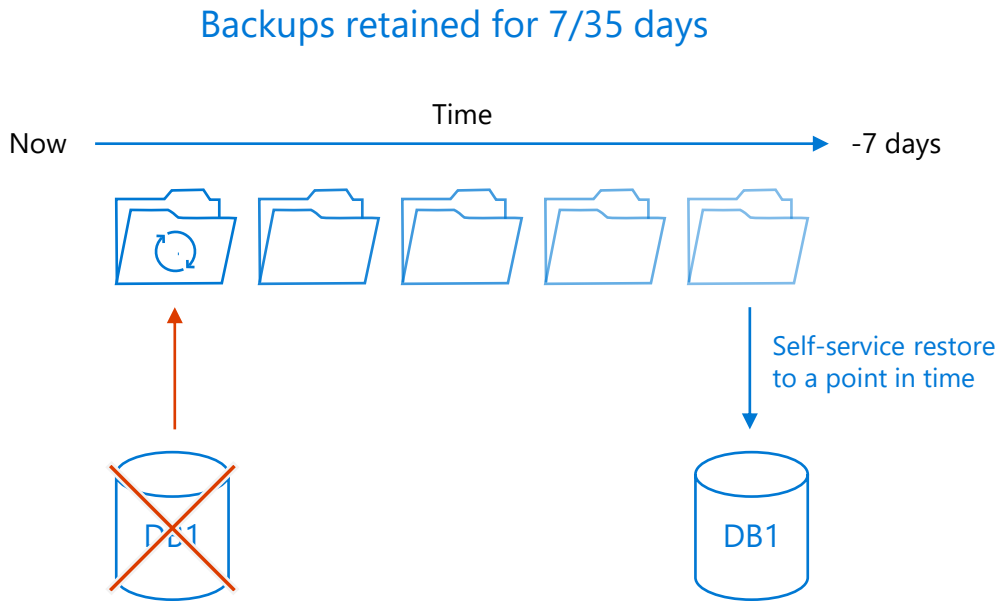
Database recovery

Restoring a deleted database

Restores the database to any point in time within the retention period

Creates a new database on the server used by the original database

You can choose to failover to the restored database or use scripts to recover data



Geo-restore protects from disaster

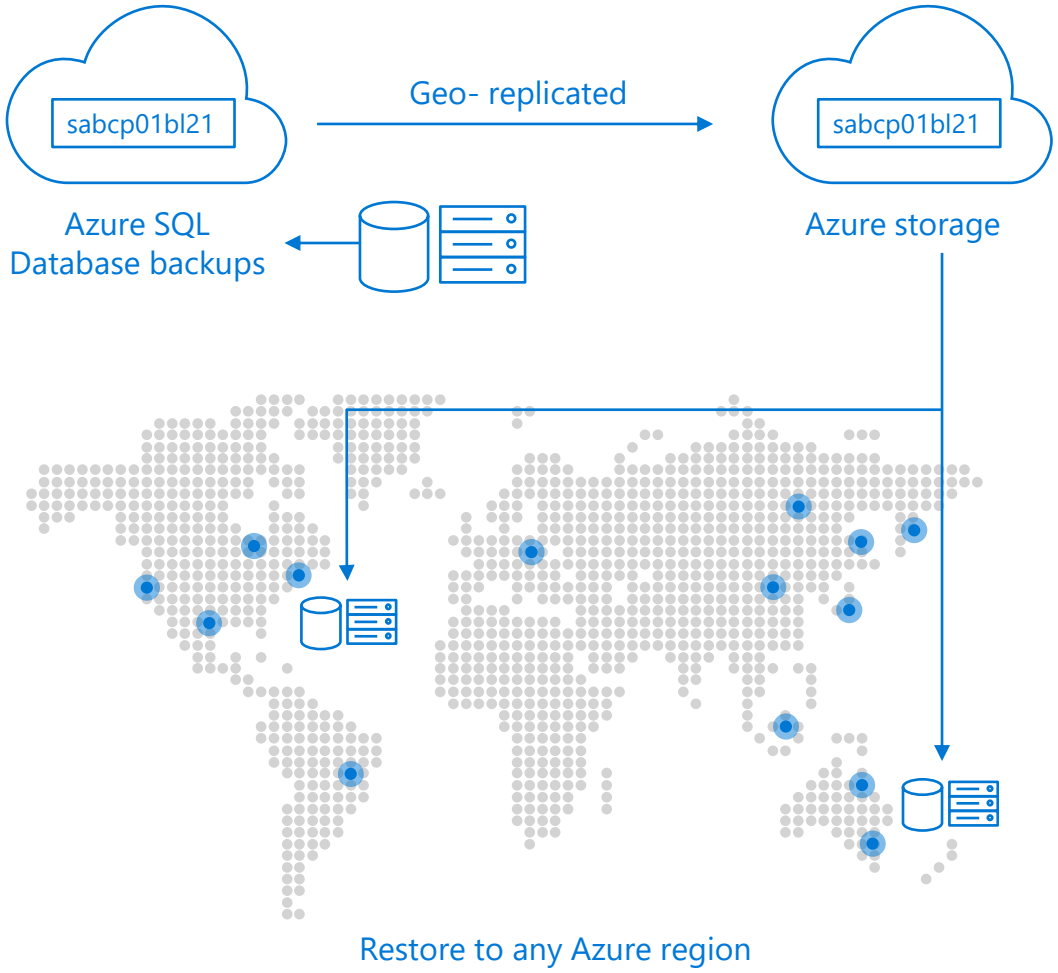
Self-service restore API

Built on geo-redundant Azure Storage

Restores last replicated backup to any Azure region as a new database

No extra cost, no capacity guarantee

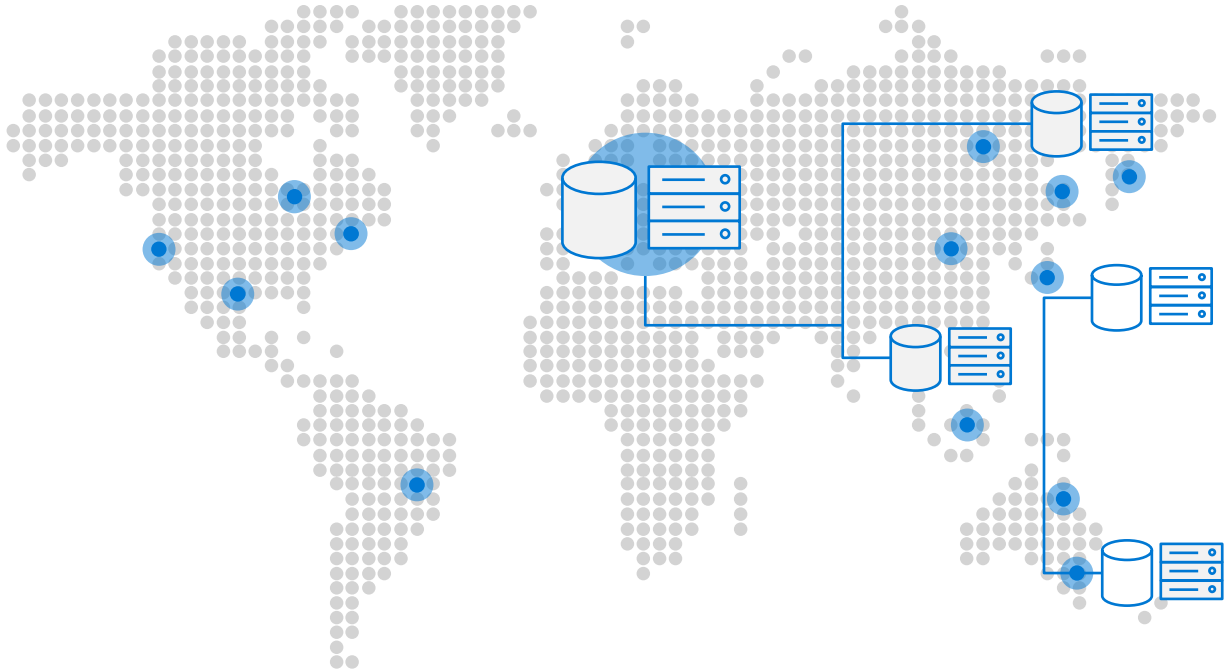
RTO≥24h, RPO=1h



Active geo-replication

Mission-critical business continuity on your terms, via programmatic APIs

Service levels	All
Readable secondaries	Up to 4
Regions available	Any Azure region
Replication	Automatic, asynchronous
Manageability tools	REST API, PowerShell, or Azure Portal
Recovery time objective (RTO)	<30 sec
Recovery point objective	<5sec
Failover	On demand



Up to 4 secondaries

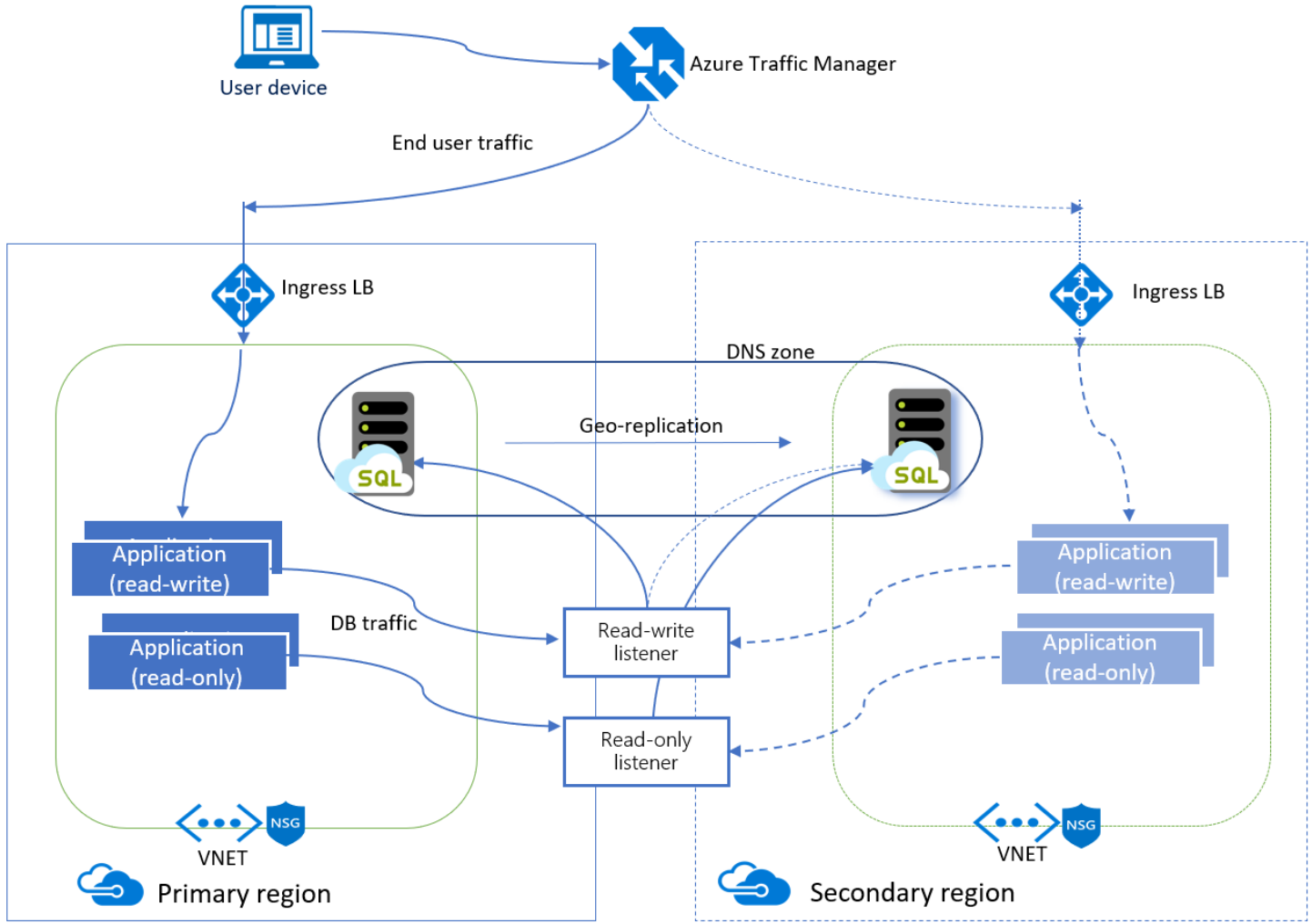
Failover groups with Managed Instance

Enable geo-replication for a group of databases or pools to another region

Automatic or manual failover policy

Read-only and read-write listener endpoints for transparent failover

Available for all service tiers



Automated administration allows you to do your job

Focus on building apps instead of management tasks

Active geo-replication provides the richest business continuity solution with the least risk of data loss and the most rapid recovery time

Extends standard geo-replication with up to four readable secondary databases in the same or different data center locations (regions)

Secondary databases can also be used for load balancing or to provide low-latency access to replicated data anywhere in the world

Automatically manage geo-replication relationship, connectivity, and failover at scale

Available in auto-failover policy or manual activation

SQL Data sync

Bi-directionally synchronizes data across multiple SQL databases and SQL Server instances

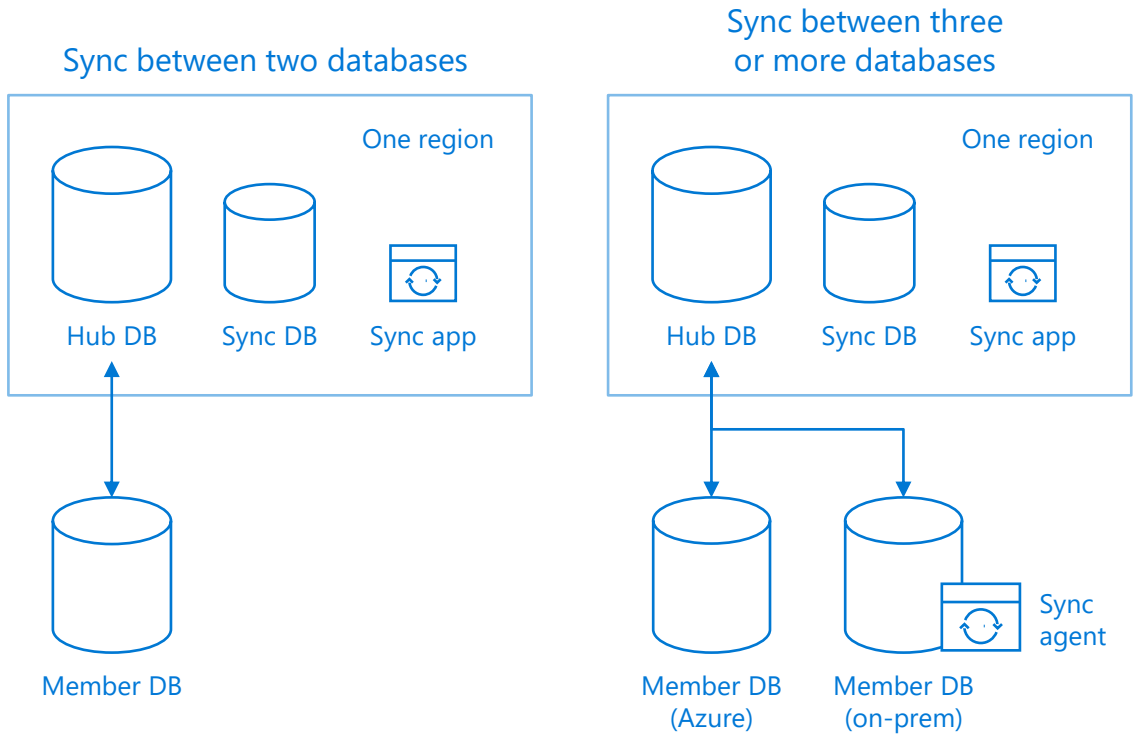
Sync occurs between the defined Hub database and individual member databases

Use cases:

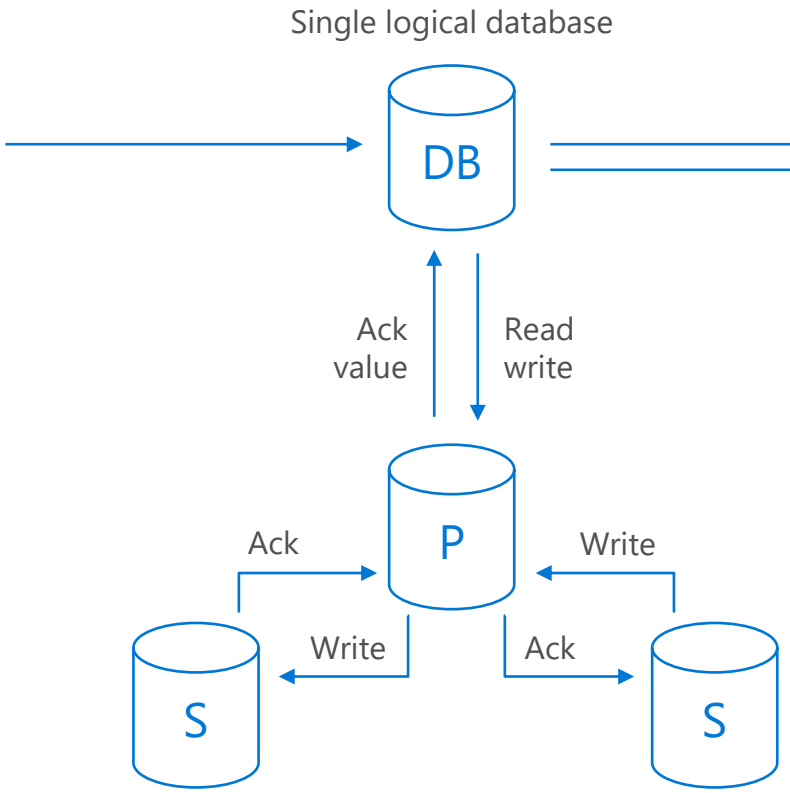
Distributed Applications

Globally Distributed Applications

Hybrid Data Synchronization

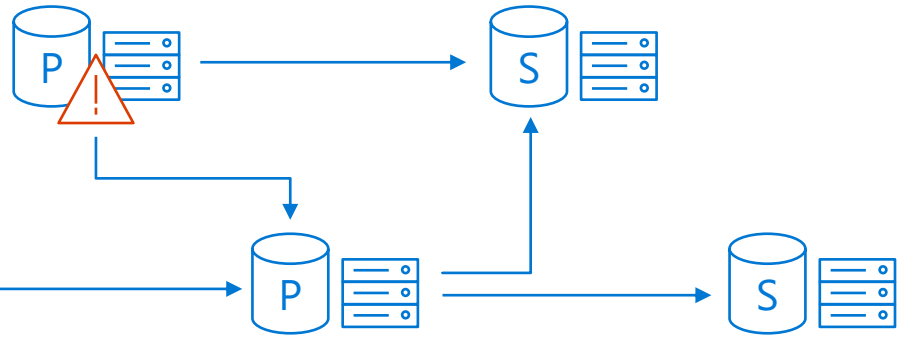


High-availability platform



Reads are completed at the primary

Writes are replicated to secondaries



Recovery from machine failure

Critical capabilities:

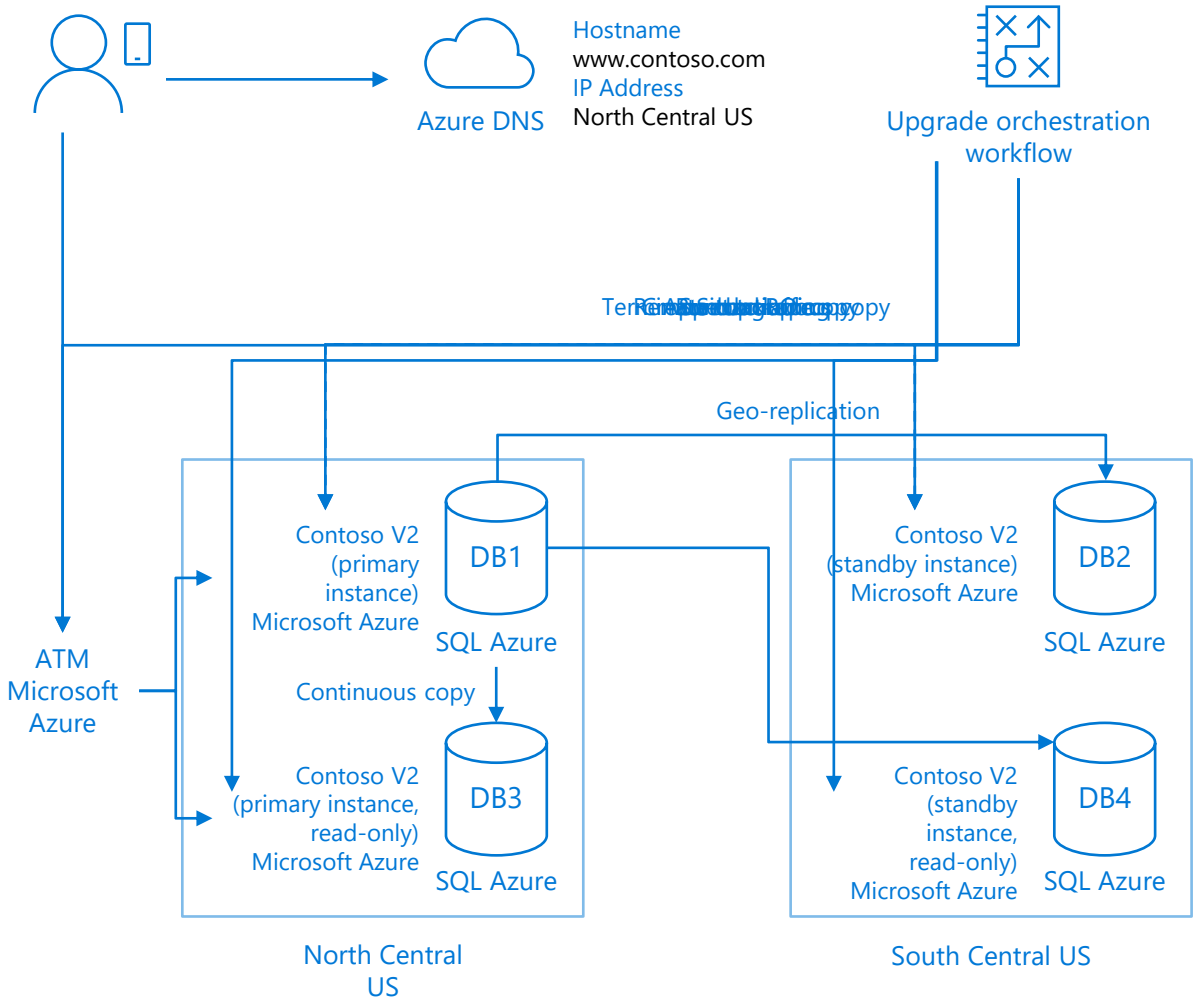
- ✓ Create new replica
- ✓ Synchronize data
- ✓ Stay consistent
- ✓ Detect failures
- ✓ Failover
- ✓ 99.99% availability

Online service upgrade

Both main copy and backup copy are protected at all times

No data loss during the upgrade process

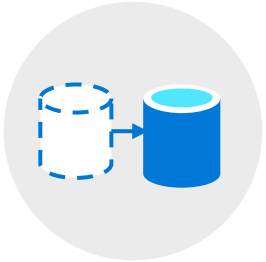
The read-only period depends on the duration of the database upgrade



Self-managed

Built-in maintenance

Remove virtually all infrastructure maintenance with Azure SQL Database, which provides automatic software patching as part of the service.

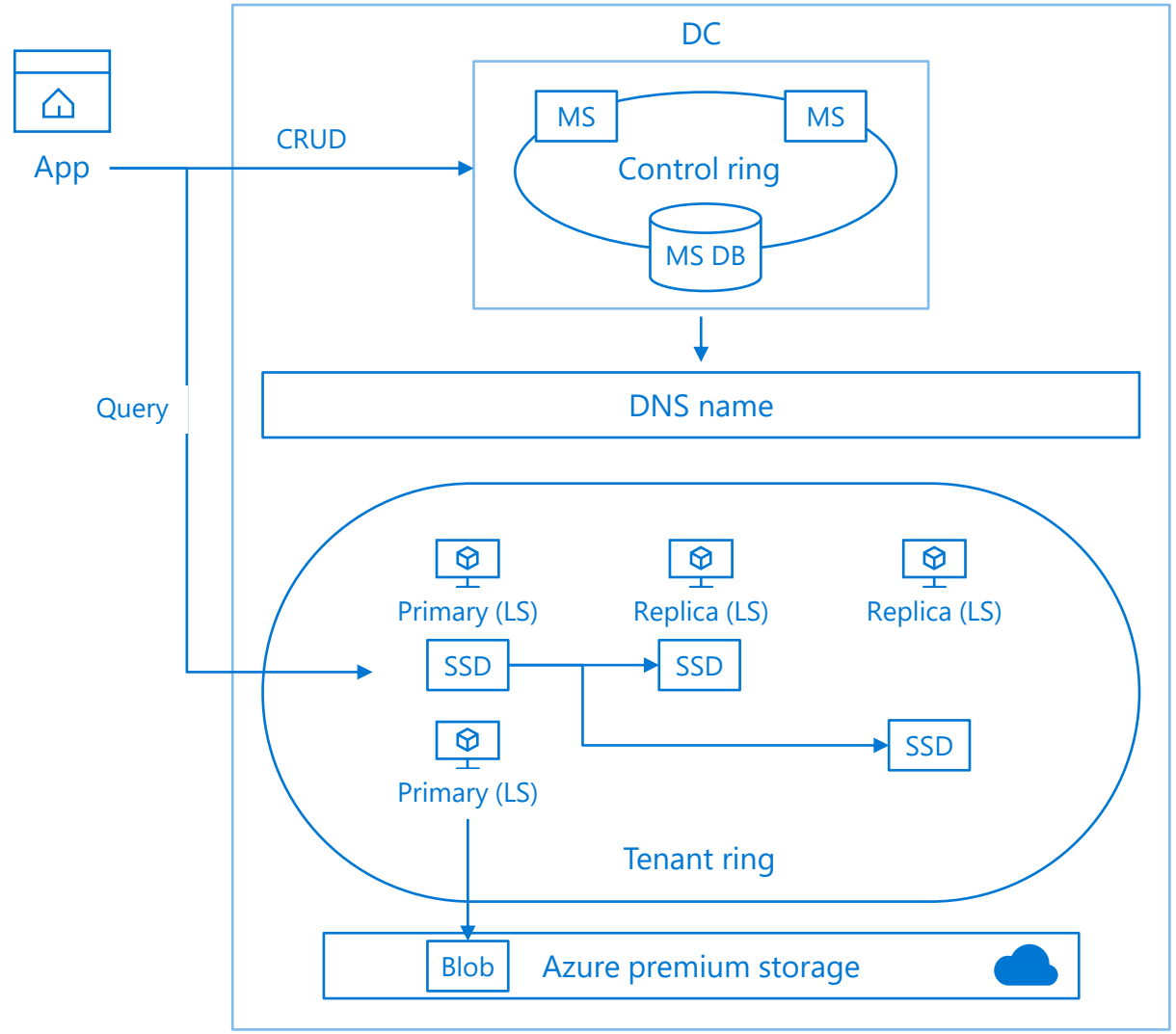


Fault tolerance

Built-in system replicas help deliver inherent data protection and database uptime. System replicas are automatically moved to new machines as old machines fail.

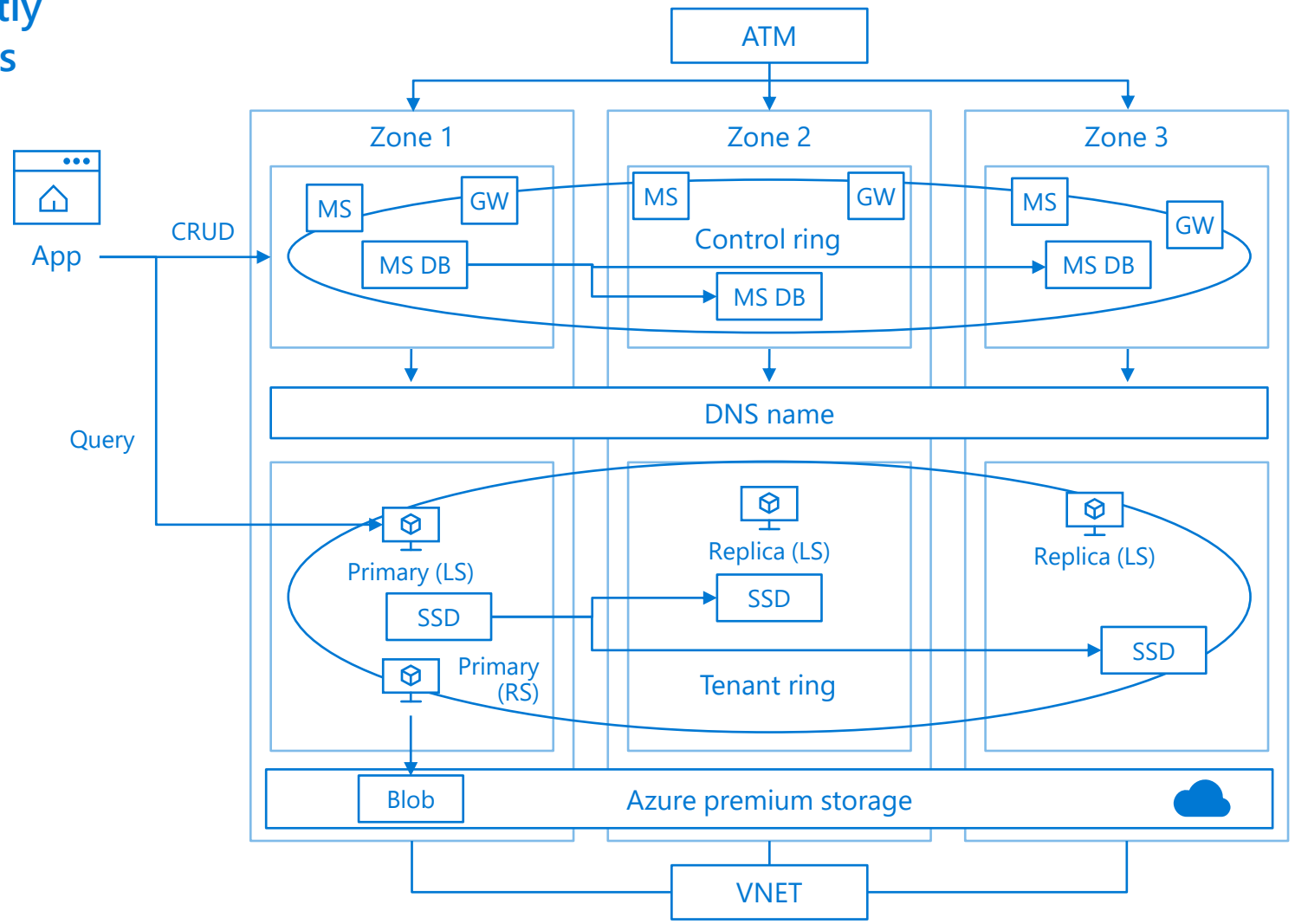
HA architecture for remote storage configurations

Remote storage configurations are used for Basic, Standard or General Purpose tiers



HA architecture for Zone redundant configuration

Zone redundant configurations are currently only supported in the Premium or Business Critical tiers



Software as a service (SaaS)

Central directory (catalog) stores customer profiles

One database per end customer (tenant)

Often for security and isolation

Some rely on schema customization

Data-dependent routing is a common data-access path

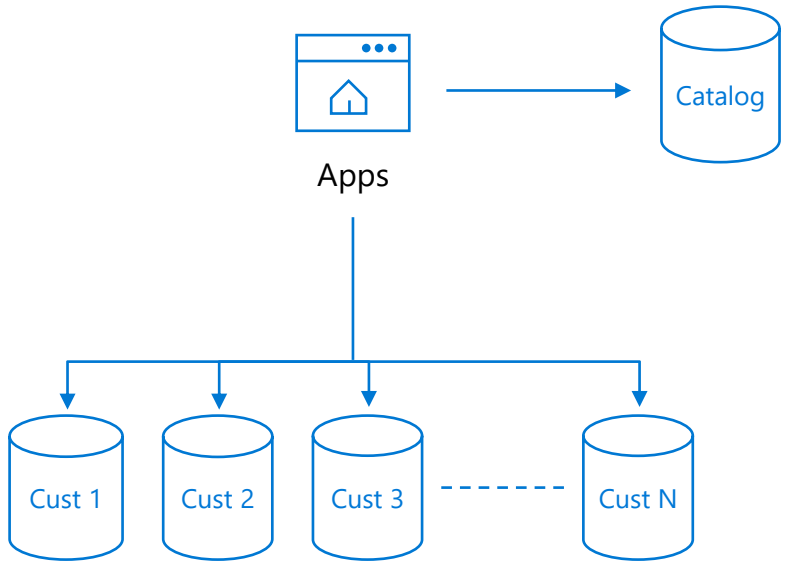
Highly selective key-lookup queries, multi-joins, and more

Mid to low data-entry rate

No need for cross-customer (fan-out) queries

Most tenants are small and "cold," and some might have hot spots

Optimize cost of goods sold (COGS) by picking the right service tier for the customer (mostly basic)



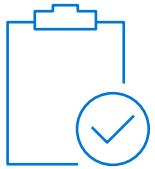
Elastic databases at a glance

Manage and monitor multi-tenant apps with the isolation benefits of one customer per database

Free yourself from the administration overhead of designing, buying, building, and managing each customer's environment

Elastic database pools and elastic database pricing model

Elastic database tools: client library and split-merge service



Elastic database job

Elastic database queries (preview) and transactions

Manage and scale multiple databases

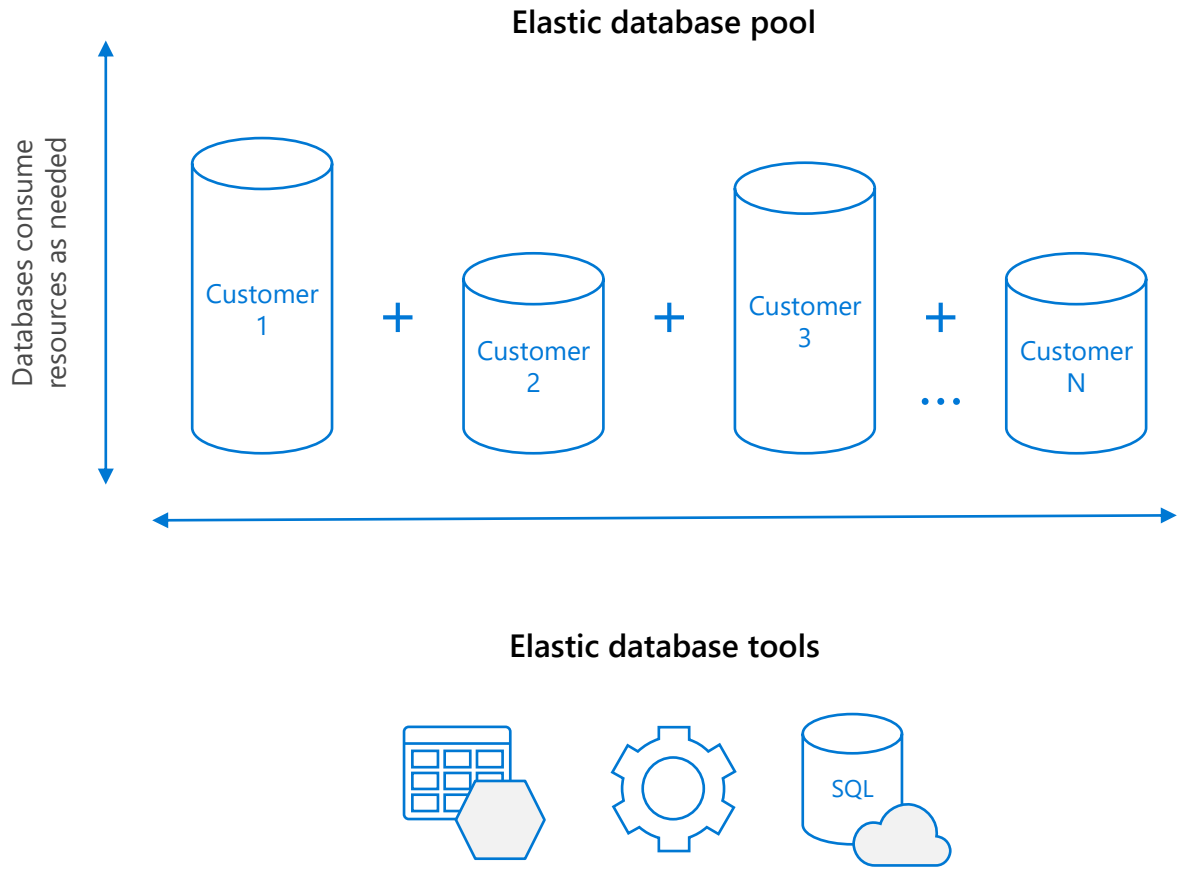
An elastic database pool is a collection of resources and storage that are used by multiple databases

Elastic database jobs allow you to perform tasks across databases in the pool, including:

Performing administrative tasks, such as deploying new schemas

Updating reference data, such as making product information common across all databases

Rebuilding indexes to improve query performance



Build secure apps

Flexibility to work your way

Platforms

Develop with your choice of popular platforms including Windows, Linux, and Mac

Tools

Use Azure Management Portal with HTML5 support, PowerShell, REST APIs, SQL Server Management Studio, Azure Data Studio and Visual Studio

Languages

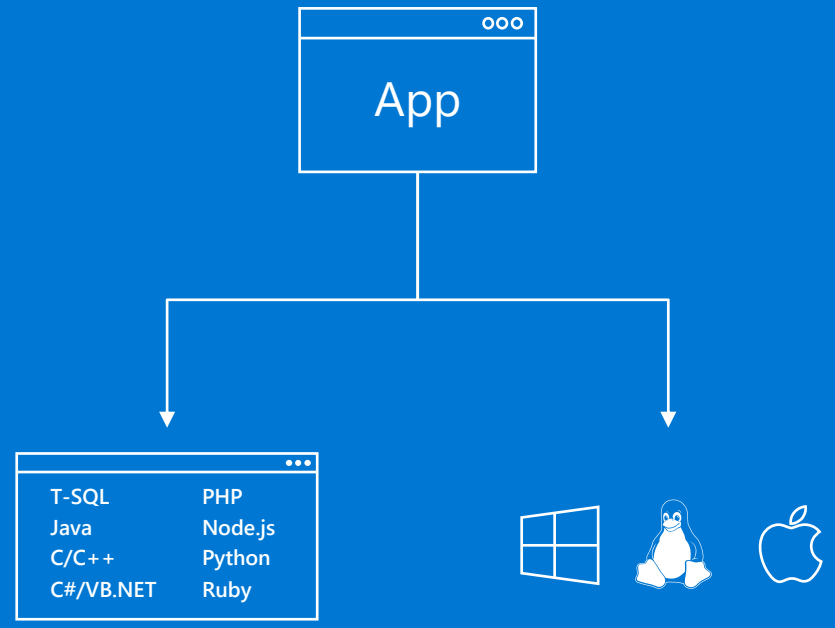
Develop with a choice of popular languages, such as C#, Java, Node.js, Ruby, PHP, and Python

Frameworks

Leverage popular frameworks including Entity, Hibernate ORM, Ruby on Rails, and Django

Your Azure solution

Build secure apps that connect with the languages and platforms you prefer



Programmatic capabilities

Platforms

Develop with a choice of popular platforms and technologies, including .NET, Java, PHP, Ruby on Rails, and Node.js

JSON support

Query JSON using standard T-SQL syntax; use JSON with all components of Azure SQL Database, such as In-Memory OLTP

Temporal tables

Keep a full history of all changes made to table; use for easy point-in-time restore or calculate trends over time

T-SQL editor

Use the HTML5-based T-SQL editor in Azure Management Portal

JSON Support in Azure SQL Database

Easily work with JSON data and integrate your database with modern services

Built-in functions
ISJSON, JSON_VALUE,
JSON_MODIFY, JSON_QUERY

```
[
 {
  “Number”:“SO43659”,
  “Date”:“2011-05-31T00:00:00”
  “AccountNumber”:“AW29825”,
  “Price”:59.99,
  “Quantity”:1
 }
 {
  “Number”:“SO43661”,
  “Date”:“2011-06-01T00:00:00”
  “AccountNumber”:“AW73565”,
  “Price”:24.99
  “Quantity”:3
 }
]
```

OPENJSON
Transforms JSON text to table

FOR JSON
Formats result set as JSON text

Number	Date	Customer	Price	Quantity
SO43659	2011-05-31 T00:00:00	MSFT	59.99	1
SO43661	2011-06-01 T00:00:00	Nokia	24.99	3

Formatting data as JSON

Format your database content as JSON directly
in a SQL query

```
SELECT CustomerName, PhoneNumber, FaxNumber
FROM Sales.Customers
FOR JSON PATH
[
{
"CustomerName":"Eric Torres",
"PhoneNumber":"(307) 555-0100",
"FaxNumber":"(307) 555-0101“
},
{
"CustomerName":"Cosmina Vlad",
"PhoneNumber":"(505) 555-0100",
"FaxNumber":"(505) 555-0101“
},
{
"CustomerName":"Bala Dixit“
,"PhoneNumber":"(209) 555-0100",
"FaxNumber":"(209) 555-0101“
}
]
```

Querying JSON data

JSON functions let you use JSON data in any SQL query

Id	Data
1	{"Price":50, "Color":"White", "tags":["toy","children","games"]}

```
SELECT Id, JSON_VALUE(Data, '$.Color'),
JSON_QUERY(Data, '$.tags')
FROM Products
WHERE JSON_VALUE(Data, '$.Color') = 'White'
```

1	White	["toy","children","games"]

Modifying JSON data

JSON functions let you use JSON data in any SQL query

Id	Data
1	{"Price":50, "Color":"White", "tags":["toy","children","games"]}

```
UPDATE Products
SET Data = JSON_MODIFY(Data, '$.Price', 60)
WHERE Id = 1
```

Id	Data
1	{"Price":**60**, "Color":"White", "tags":["toy","children","games"]}

Temporal Tables

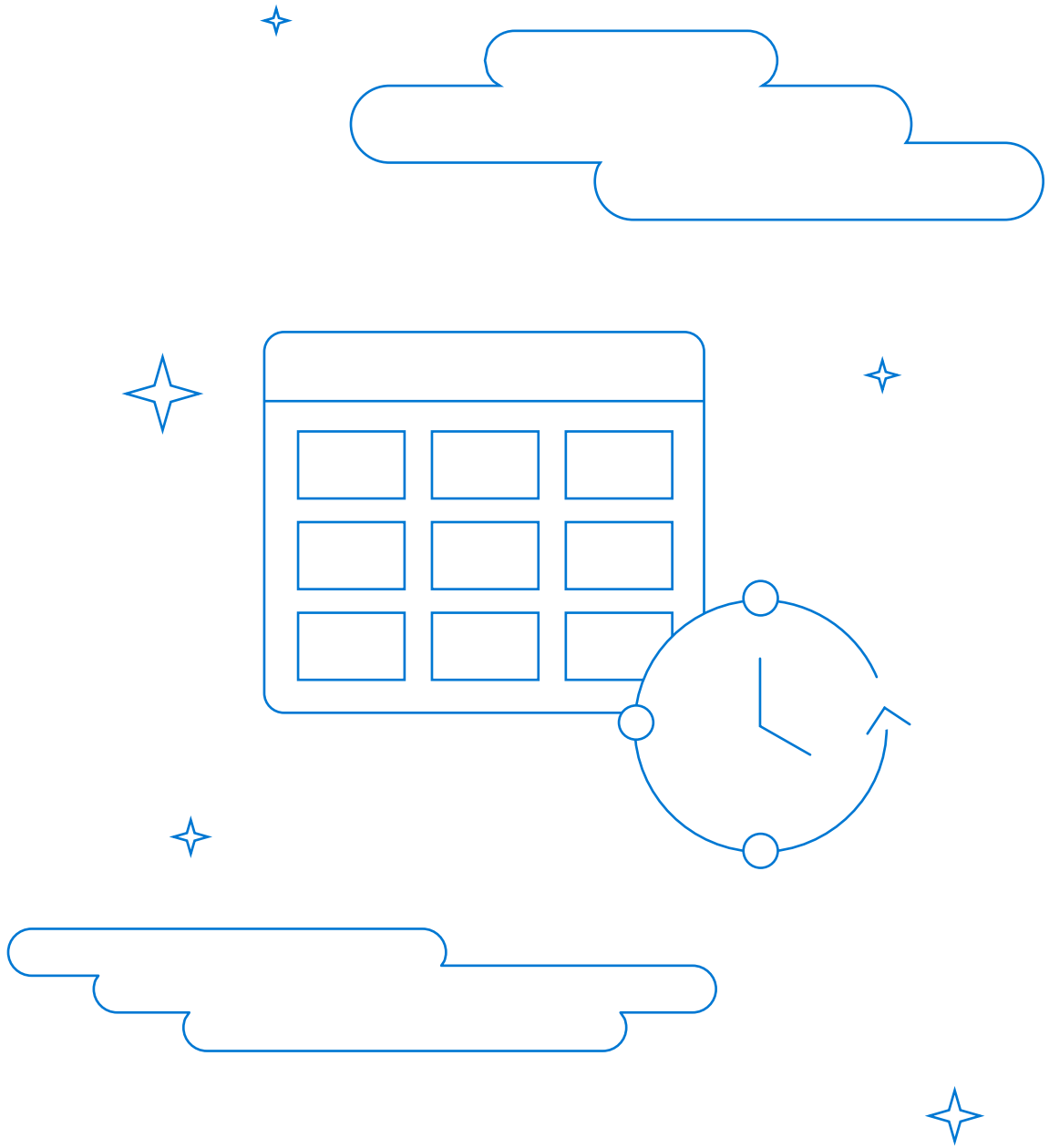
Track and analyze the history of changes in data with no custom coding

Audit data changes

Reconstruct state of data stored in the table at any point in time

Calculate trends over time

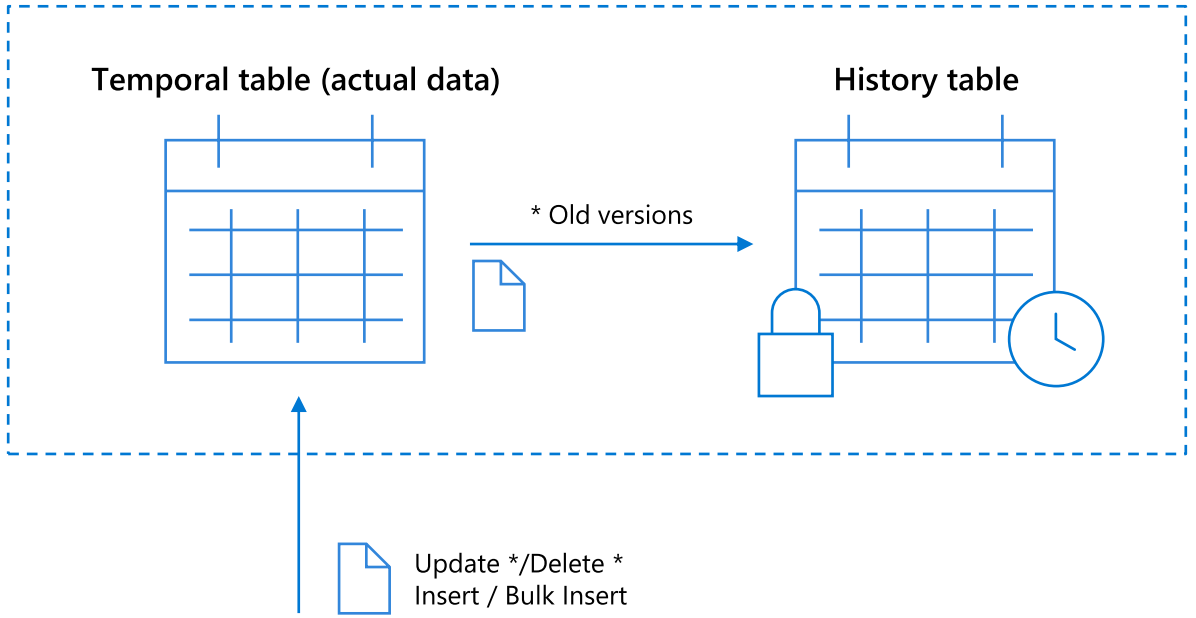
Maintain a slowly changing dimension for decision support applications



Reveal historical data with temporal tables

Implemented as a pair of tables—one current, one historical

The history table cannot have data inserted or deleted from it directly and its schema cannot be directly modified



Creating a temporal table

Temporal Tables can be created with your preferred tool

SQL Server Management Studio

SQL Server Data Tools

T-SQL

```
CREATE TABLE WebsiteUserInfo
(
    [UserID] int NOT NULL PRIMARY KEY CLUSTERED
  , [UserName] nvarchar(100) NOT NULL
  , [PagesVisited] int NOT NULL
  , [ValidFrom] datetime2 (0) GENERATED ALWAYS AS ROW START
  , [ValidTo] datetime2 (0) GENERATED ALWAYS AS ROW END
  , PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)
 )
 WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE =
dbo.WebsiteUserInfoHistory));
```

Reading from the history table

There are many ways to read data from the history table

Example) To see the top 10 users ordered by the number of visited web pages as of an hour ago:

Use the AS OF clause

```
DECLARE @hourAgo datetime2 = DATEADD(HOUR, -1, SYSUTCDATETIME());
SELECT TOP 10 * FROM dbo.WebsiteUserInfo FOR SYSTEM_TIME AS OF @hourAgo
ORDER BY PagesVisited DESC
```

Reading from the history table

Example) To perform basic statistical analysis for the previous day:

Use the BETWEEN … AND … clause

```
DECLARE @twoDaysAgo datetime2 = DATEADD(DAY, -2, SYSUTCDATETIME());

DECLARE @aDayAgo datetime2 = DATEADD(DAY, -1, SYSUTCDATETIME());

SELECT UserID, SUM (PagesVisited) as TotalVisitedPages, AVG (PagesVisited) as AverageVisitedPages,

MAX (PagesVisited) AS MaxVisitedPages, MIN (PagesVisited) AS MinVisitedPages,

STDEV (PagesVisited) as StDevVisitedPages

FROM dbo.WebsiteUserInfo

FOR SYSTEM_TIME BETWEEN @twoDaysAgo AND @aDayAgo

GROUP BY UserId
```

Setting Retention history

Temporal Tables may increase database size more than regular tables, particularly if:

- You retain historical data for a long period of time
- You have an update or delete heavy data modification pattern

Developing a data retention policy for managing data in the history table is an important aspect of planning and managing the lifecycle of every temporal table

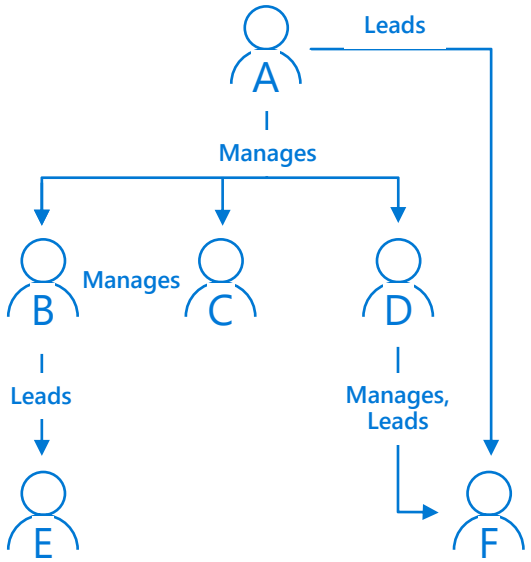
Ensure retention policy is enabled at the database level

```
ALTER DATABASE <myDB>
SET TEMPORAL_HISTORY_RETENTION ON
```

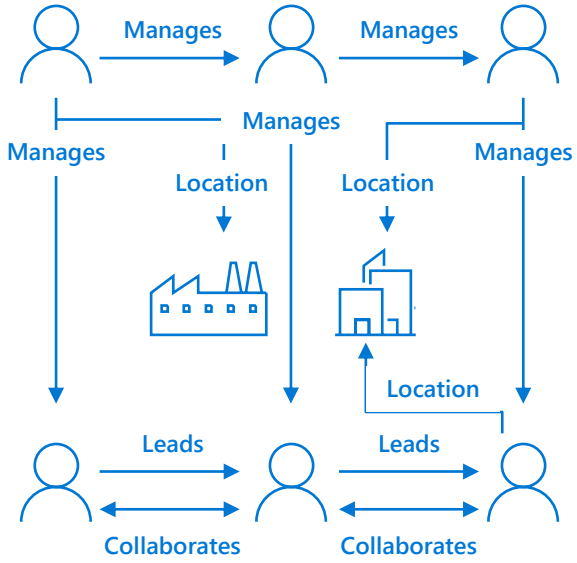
Retention policy can be set when the table is created or can be defined or altered after table creation

```
ALTER TABLE dbo.WebsiteUserInfo
SET (SYSTEM_VERSIONING = ON (HISTORY_RETENTION_PERIOD = 9 MONTHS));
```

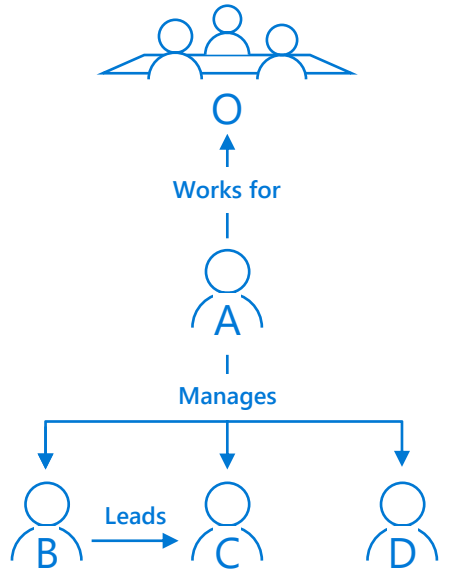
Graph Database use cases



Uncover interconnected or hierarchical data entities with multiple parents



Organically grow many-to-many relationships as the business evolves



Analyze interconnected data relationships, and identify non-obvious connections

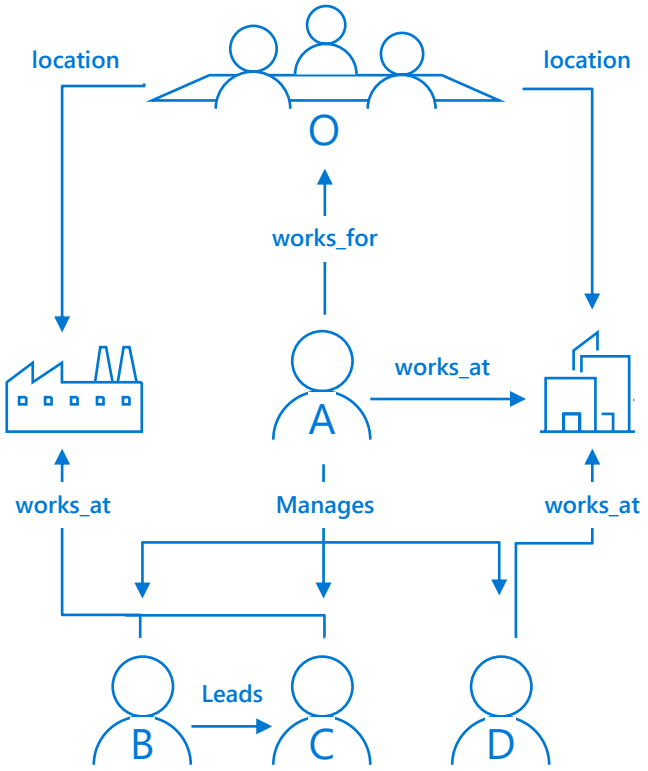
SQL Graph

Graph objects

Create nodes and edges

Properties associated with nodes and edges

```
CREATE TABLE Person (ID INTEGER PRIMARY KEY,
name VARCHAR(100)) AS NODE;

CREATE TABLE Organization (ID INTEGER PRIMARY KEY,
name VARCHAR(100)) AS NODE;

CREATE TABLE Manages AS EDGE;

CREATE TABLE works_for (StartDate date) AS EDGE;
```



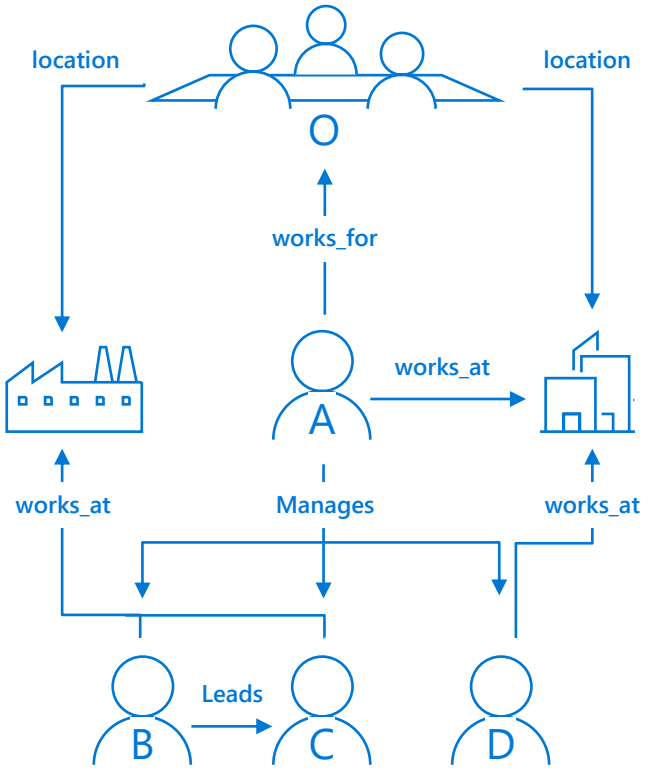
SQL Graph

Graph objects

Query language extension

Multi-hop navigation and join-free pattern matching

```
SELECT person2.name
  FROM Person person1,
       Manages,
       Person person2,
       works_at,
       location
 WHERE MATCH(person1-(Manages)->person2-(works_at)->location)
   AND person1.name = 'Alice'
```



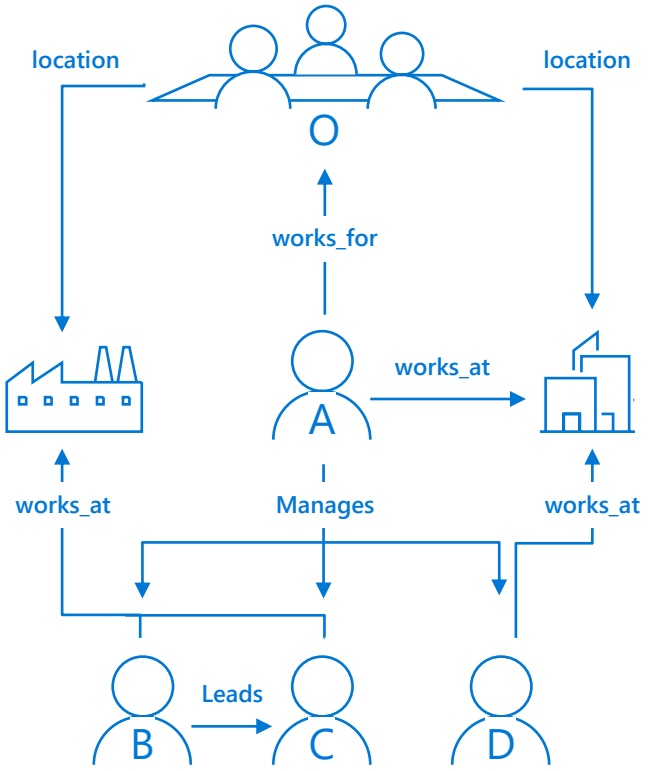
SQL Graph

Graph objects

Query language extension

Integrated in SQL Engine

Queries can lookup against existing SQL database tables and graph nodes/edges
Column store, Advanced Analytics/ML, HA, etc.
Security and compliance



SQL Graph

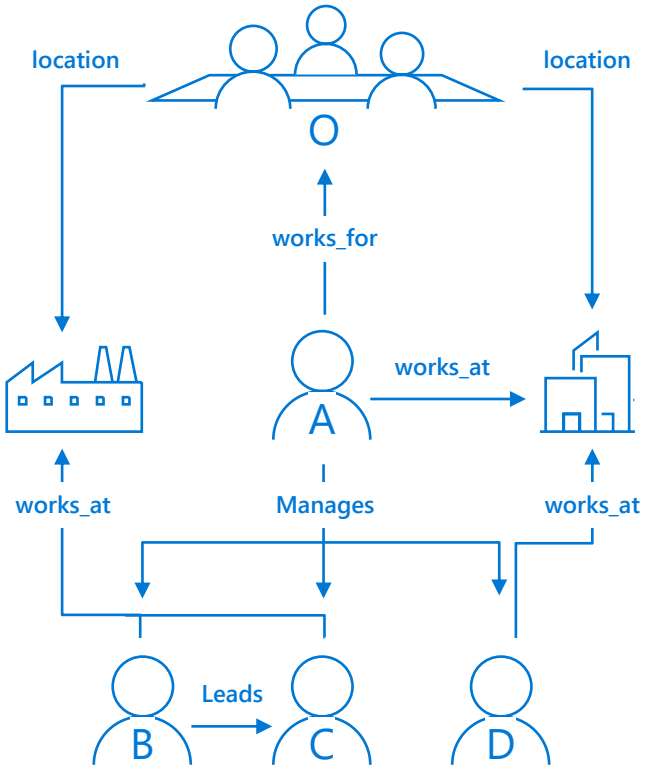
Graph objects

Query language extension

Integrated in SQL Engine

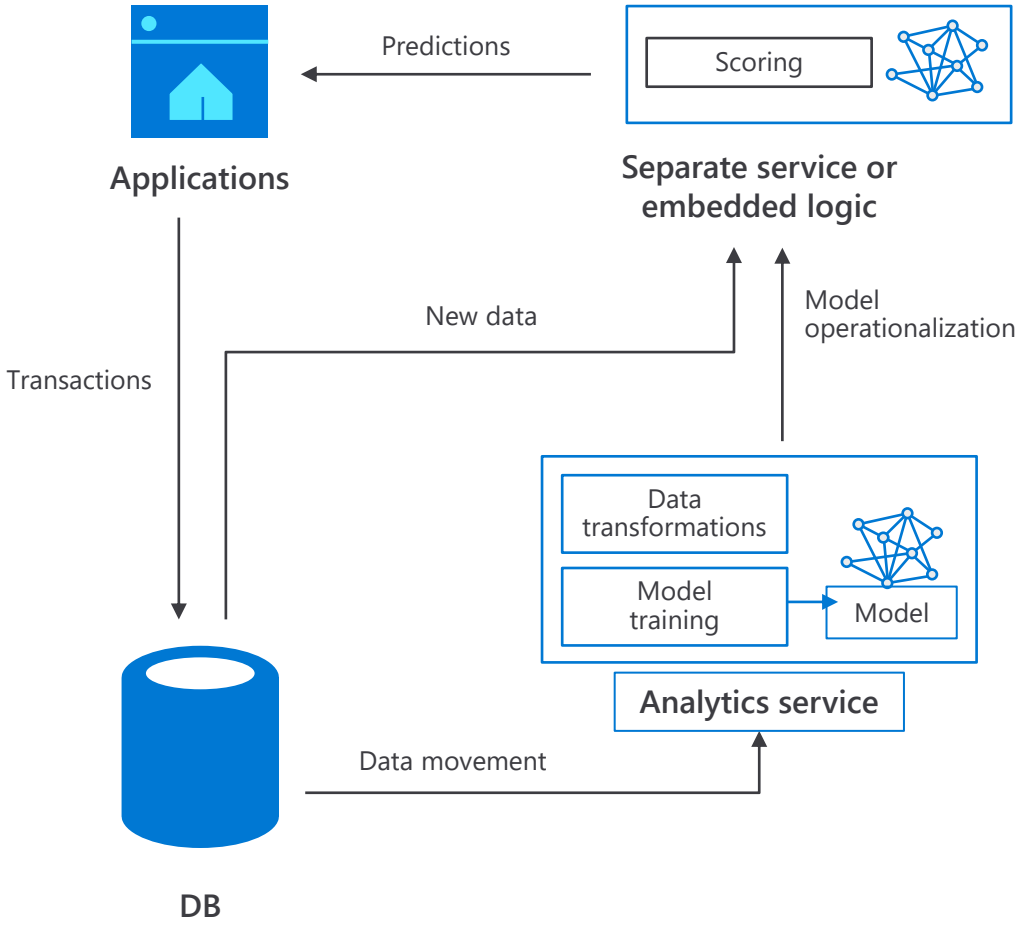
Tooling and ecosystem

Existing tools will all work out of the box, including backup and restore, import and export, etc.

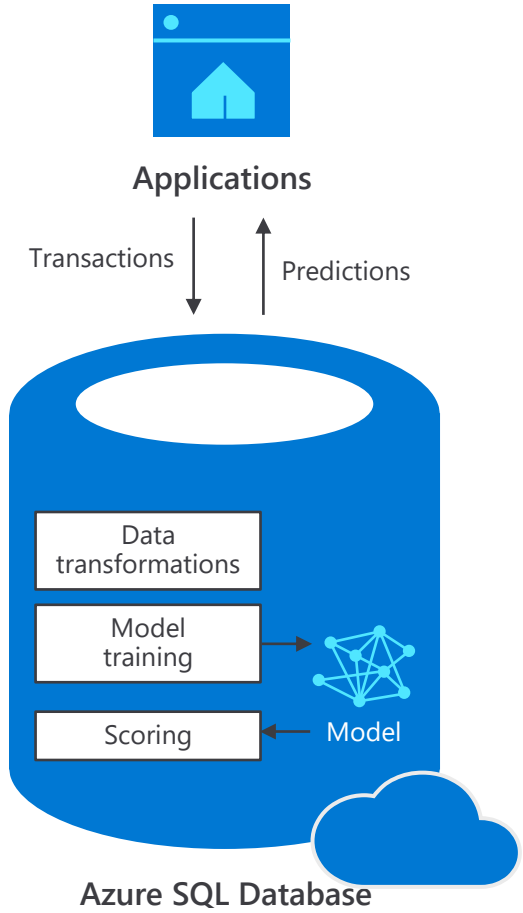


Machine Learning Services in Azure SQL Database

Machine learning outside of database



In-database machine learning



Machine Learning Services in Azure SQL Database

Capabilities

Extensible in-database analytics, exposed through T-SQL

Preview with R, Python coming soon

No data movement, resulting in faster time to insights

Real-time analytics on transactional data with native PREDICT

Integration with existing application workflows

Unified governance across analytics and storage

Running R script in Azure SQL Database:

```
/* Input table schema */
create table Iris_Data (name varchar(100), length int, width int);
/* Model table schema */
create table my_iris_model (model varbinary(max));

declare @iris_model varbinary(max) = (select model from my_iris_model);
exec sp_execute_external_script
  @language = 'R'
, @script = '
IrisPredict <- function(data, model){
 library(e1071)
 predicted_species <- predict(model, data)
 return(predicted_species)
}
IrisPredict(input_data_1, model);
'
, @parallel = default
, @input_data_1 = N'select * from Iris_Data'
, @params = N'@model varbinary(max)'
, @model = @iris_model
with result sets ((name varchar(100), length int, width int
, species varchar(30)));
```

Values highlighted in yellow are SQL queries embedded in the original R script

Values highlighted in aqua are R variables that bind to SQL variables by name

Migrate to Azure SQL Database

Database migration journey

Assess

- Involve stakeholders
- Calculate your TCO
- Discover and evaluate apps

Migrate

- Select a migration strategy
- Find recommended tools
- Apply the migration strategy

Optimize

- Analyze your costs
- Save with offers
- Reinvest to do more

Secure and manage

- Industry-leading security
- Protect your data
- Monitor cloud health

Tools and services for your migration journey

Database Migration Service (DMS)

Enables offline & online migrations to Azure SQL Database

Data Migration Assistant (DMA)

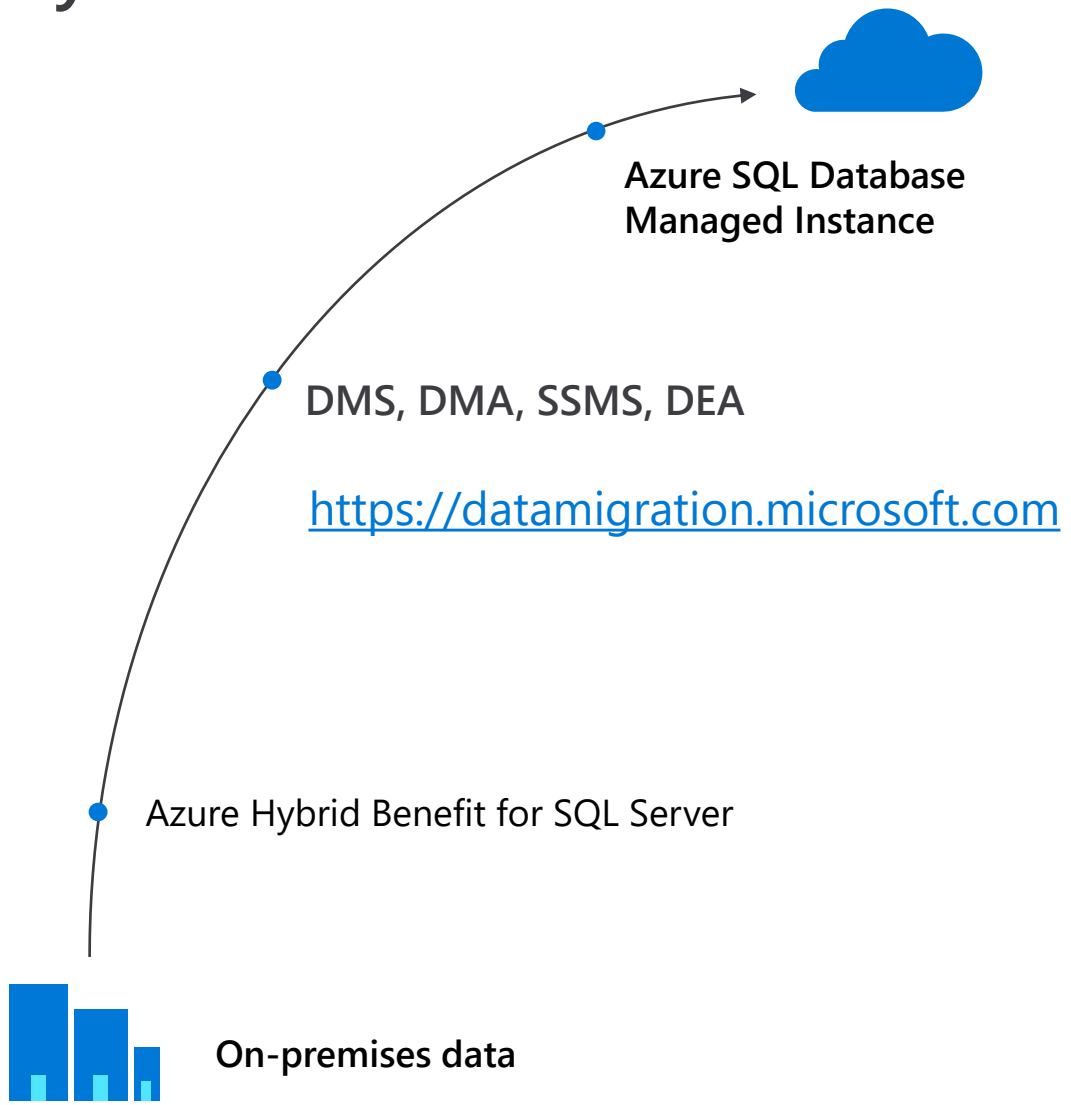
Assess database compatibility and feature parity

SQL Server Management Studio (SSMS)

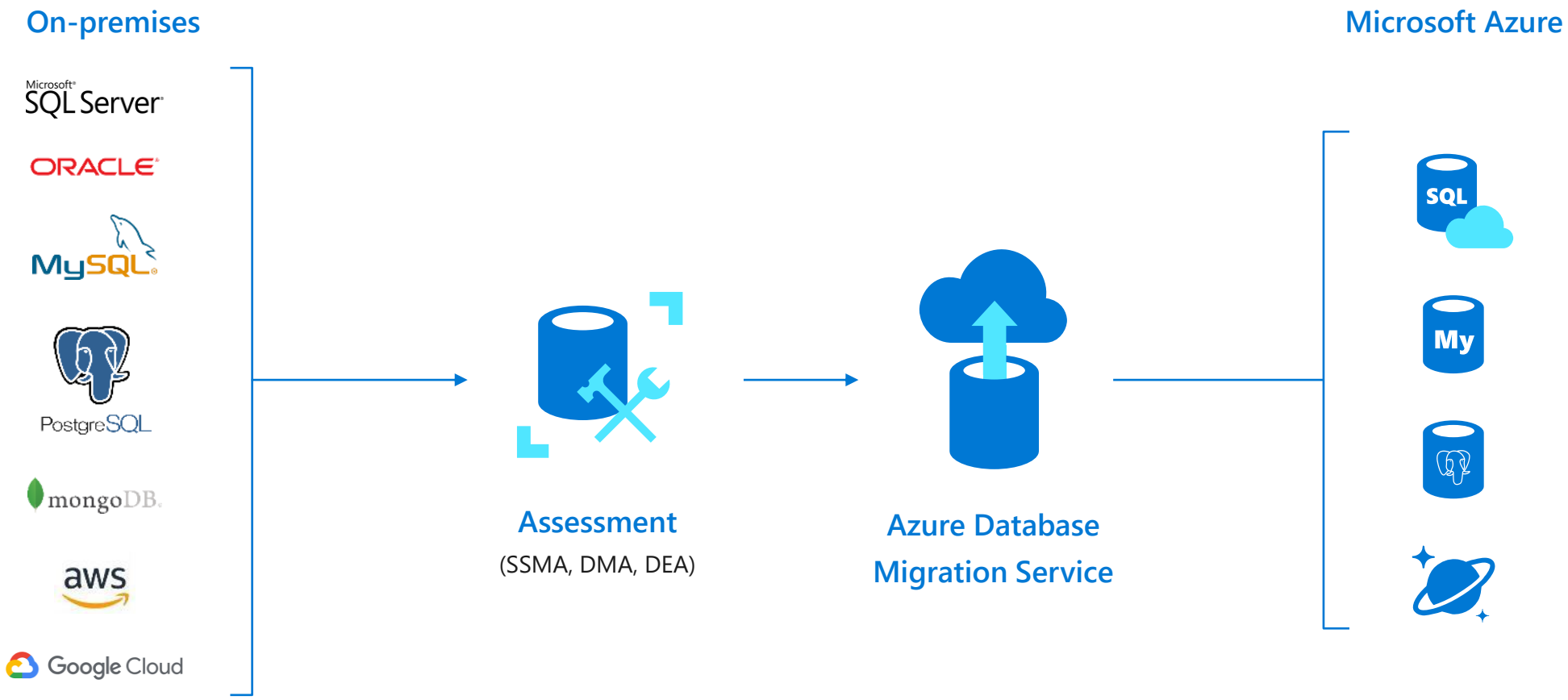
Integrated environment for managing SQL Server

Database Experimentation Assistant (DEA)

Evaluate target version of SQL Server for a given workload

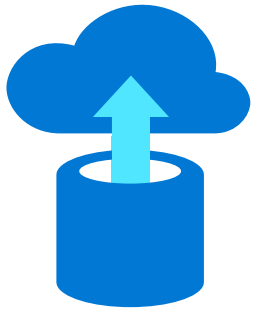


Tools and services for your migration journey

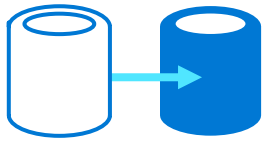


Azure Database Migration Service

Accelerate your transition to Azure

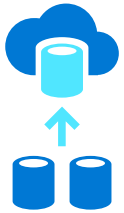


Homogeneous sources



Heterogeneous sources

Orchestration



Scale migration

Near-zero downtime

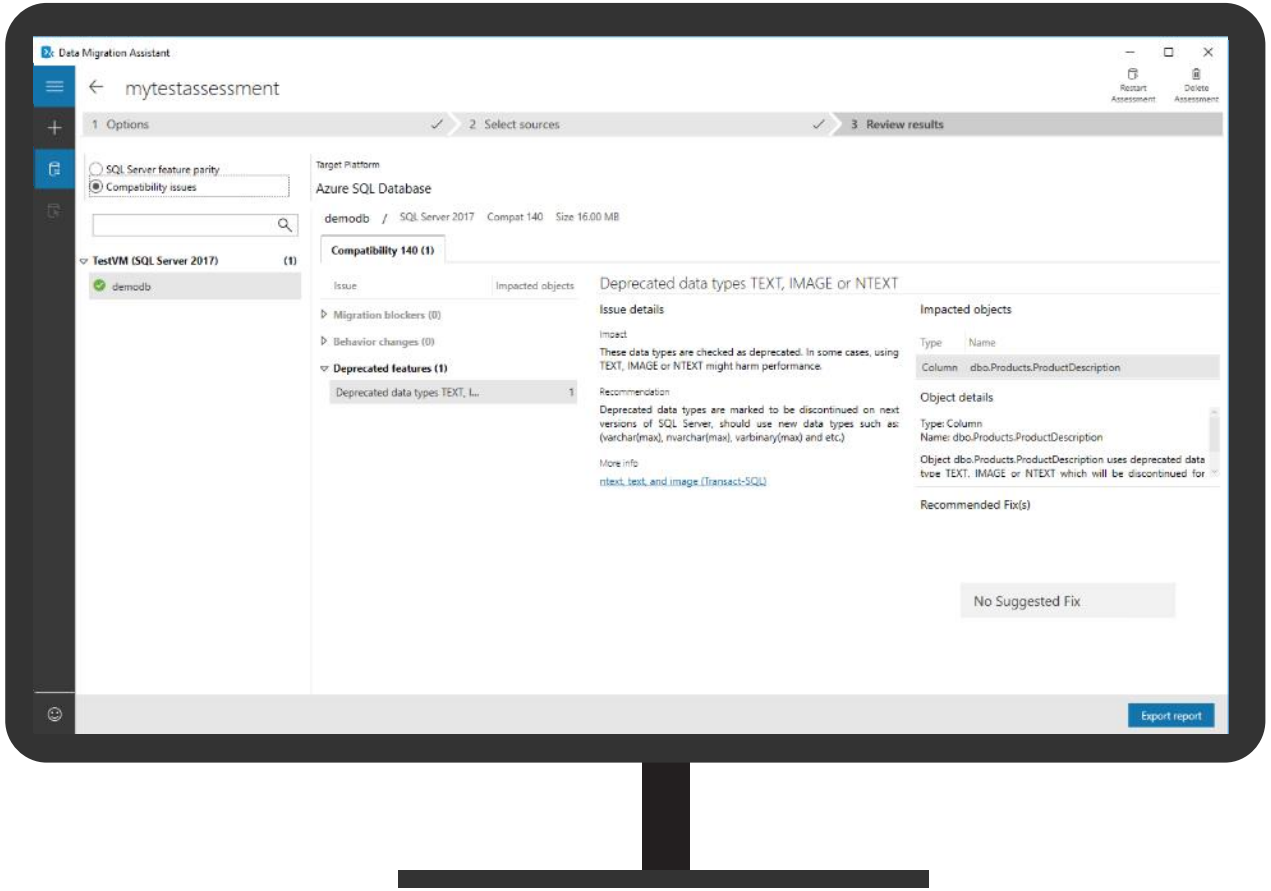
A seamless, end-to-end solution for moving on-premises databases to Azure

Data Migration Assistant

Assess on-premises SQL Server instance(s) for migrating to Azure SQL database(s)

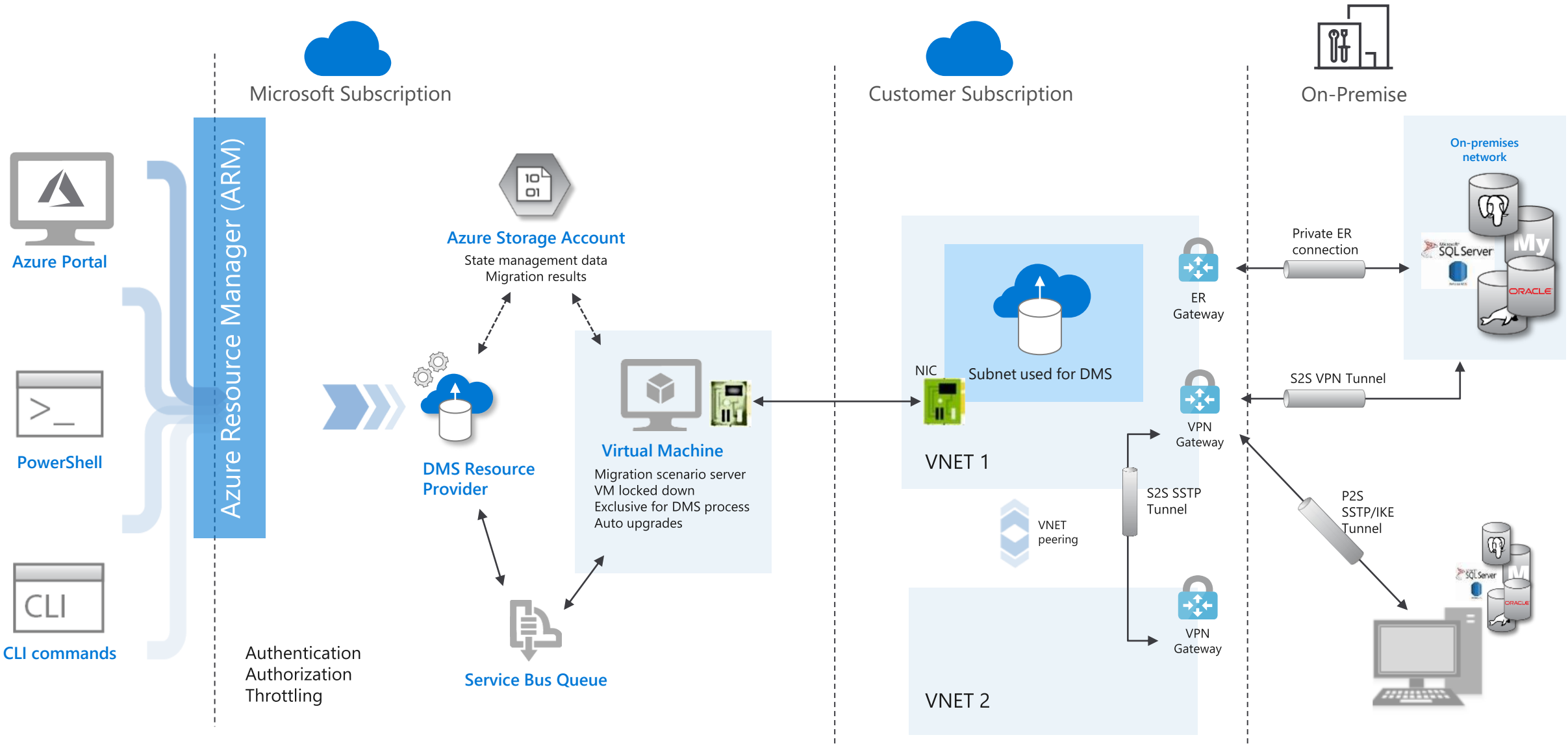
Discover issues that can affect an upgrade

Migrate an on-premises SQL Server instance to a modern SQL Server instance



Database Experimentation Assistant

Azure Database Migration Service



Online migration with backup and restore technology

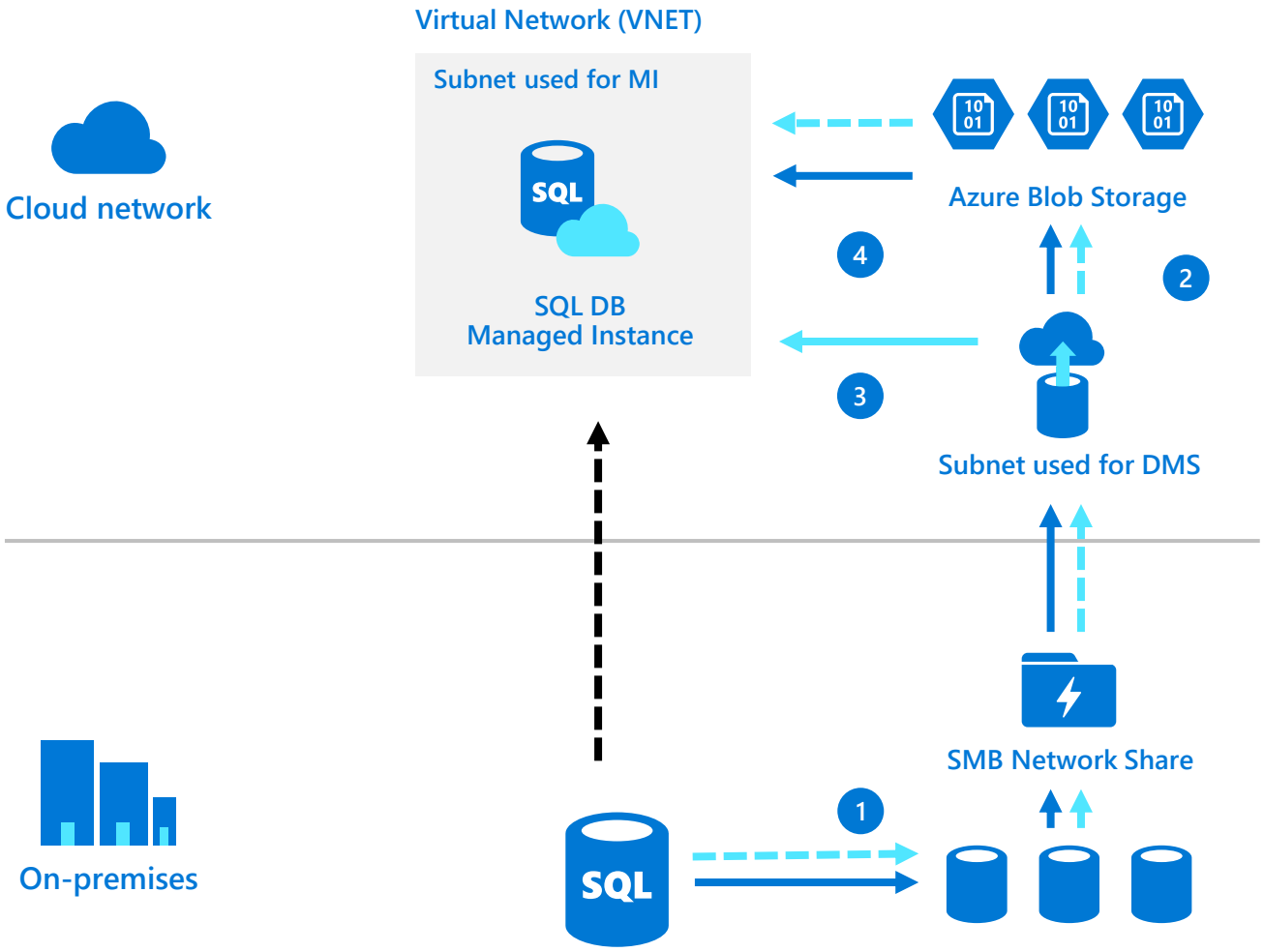
SQL Server to Azure SQL Database Managed Instance example

1. Provide existing backups in network share
2. DMS upload backup files to Azure storage
3. DMS initiate the migration to Managed Instance
4. Full backup restored and Transaction log backups continuously applied until cutover

Stop incoming traffic to source databases, provide Tail-Log backup, initiate cutover in DMS and change the application connection strings

Legend

- Full Database backup files
- Transaction log backup files
- Site to site connectivity (VPN or ExpressRoute)

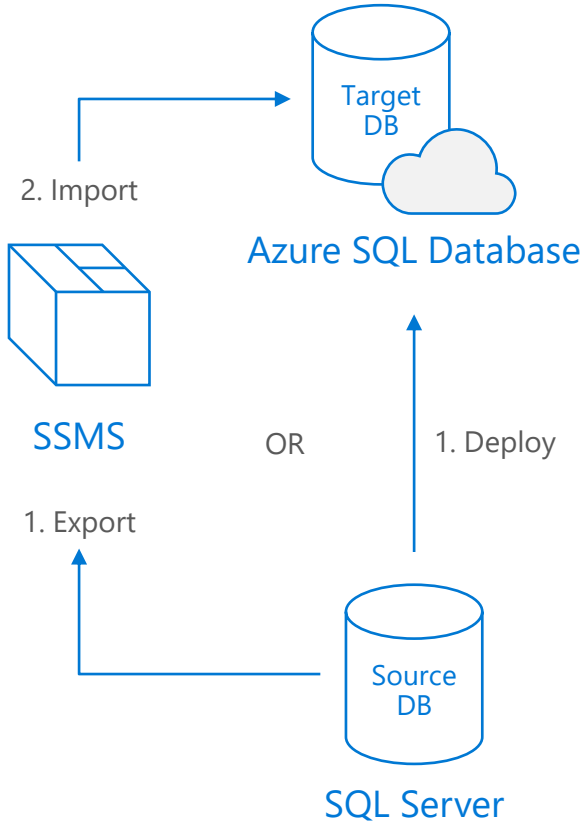


SQL server management studio

Migrate a compatible database using SQL Server Management Studio

Use SSMS to deploy to Azure SQL Database

Use SSMS to export a BACPAC and then import it to Azure SQL Database



Migration Cookbook

Migrate an on-premises SQL Server database to Azure SQL Database

The Migration Cookbook describes various approaches you can use to migrate an on-premises SQL Server database to the latest Azure SQL Database Update

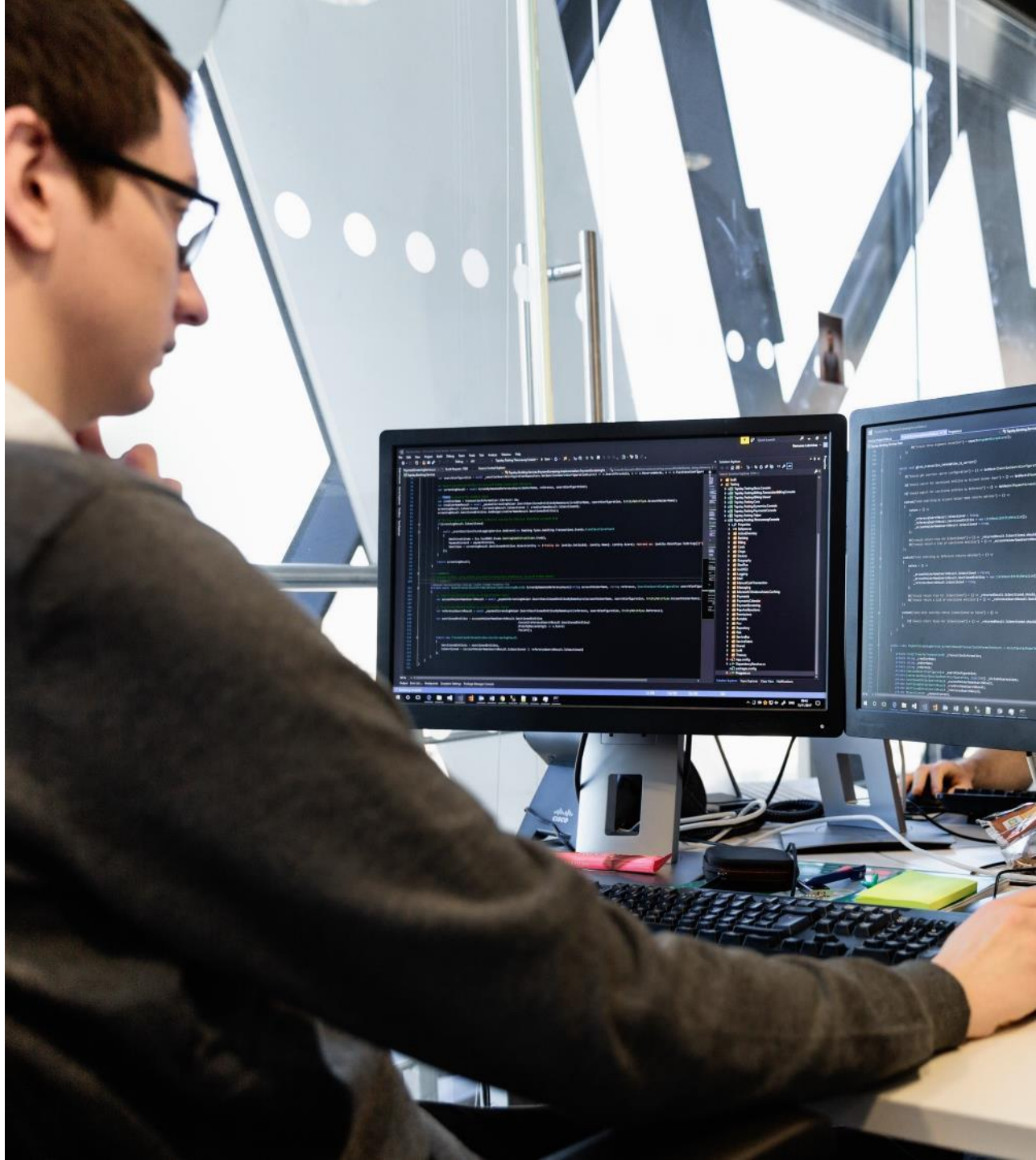
Download: https://azure.microsoft.com/en-us/resources/choosing-your-database-migration-path-to-azure/en-us/

Migration Centre: https://azure.microsoft.com/en-us/migration/

Secures and protects your application data

Learning Objectives

Always encrypted
Transparent data encryption
Vulnerability Assessment
Information Protection
Row-level security
Dynamic data masking
Threat detection
Auditing and compliance



Layers of protection

Azure Active Directory

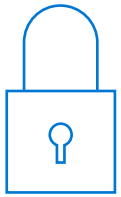
Centrally manage and control identity and user access

Encryption

Encrypt a database, associated backups, and log files at rest—without changing your app

Data protection

Protect data at rest, in motion, or in use



Row-Level Security

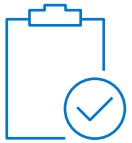
Control which users can access specific row-level data

Auditing and threat detection

Get notified of potential threats with auditing tools and anomalous activity alerting

Regulatory compliance

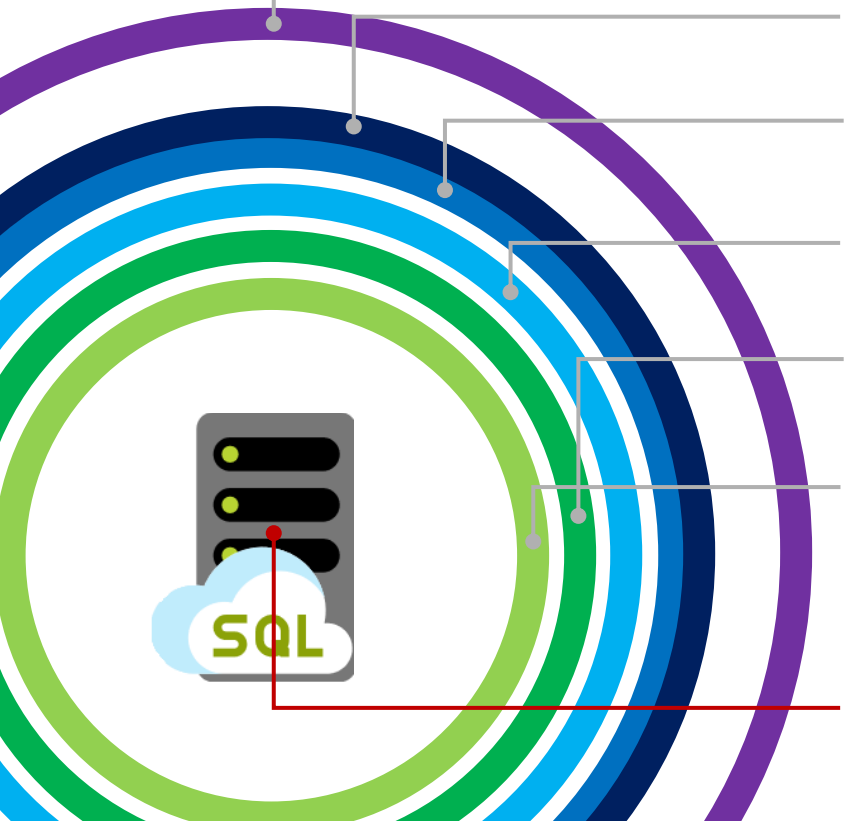
Leverage ISO/IEC 27001/27002, Fed RAMP/FISMA, SOC, HIPPA, and PCI DSS compliance



Vulnerability Assessment and Data Discovery & Classification

Discover and classify sensitive data and discover and remediate insecure configurations.

Enterprise-grade security that is easy to use



PHYSICAL SECURITY

NETWORK SECURITY

CLUSTER SECURITY

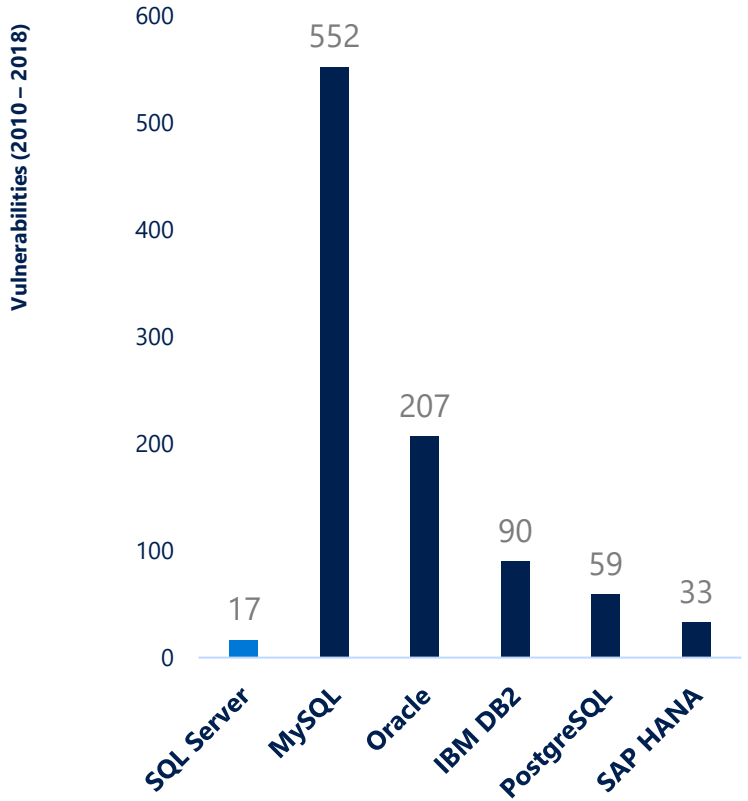
ACCESS MANAGEMENT

THREAT PROTECTION

INFORMATION PROTECTION

CUSTOMER DATA

Trusted: most secure over last 7 years



Azure Active Directory authentication

Overview

Manage user identities in one location

Enable access to Azure SQL Database and other Microsoft services with Azure Active Directory user identities and groups

Benefits

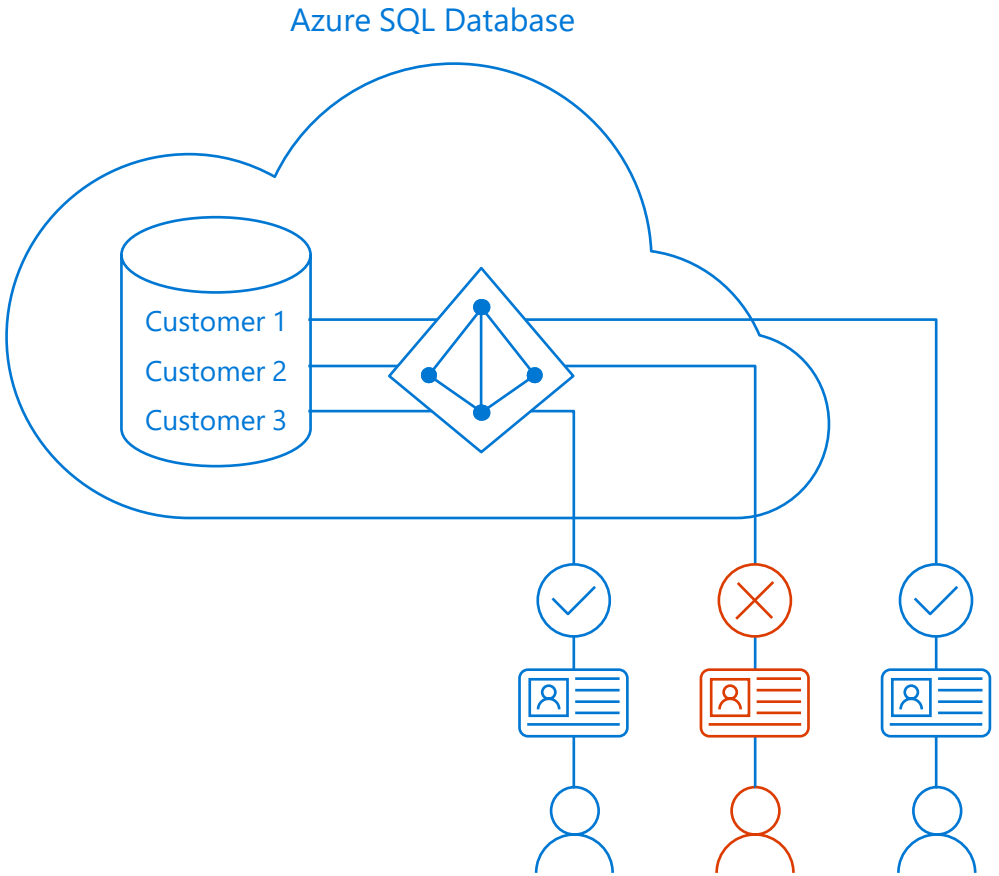
Alternative to SQL Server authentication

Limits proliferation of user identities across databases

Allows password rotation in a single place

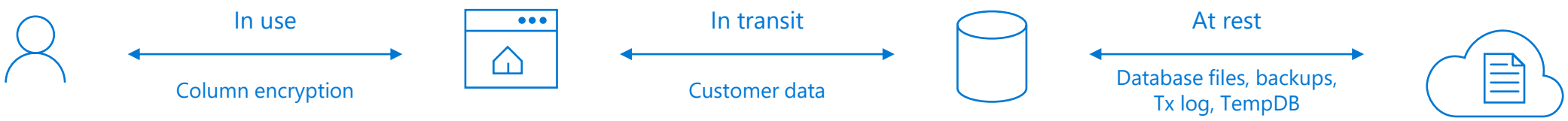
Enables management of database permissions by using external Azure Active Directory groups

Eliminates the need to store passwords



Types of data encryption

Data encryption	Encryption technology	Customer value
In transit	Transport Layer Security (TLS) from the client to the server	Protects data between client and server against snooping and man-in-the-middle attacks *Azure SQL Database is phasing out Secure Sockets Layer (SSL) 3.0 and TLS 1.0 in favor of TLS 1.2
At rest	Transparent Data Encryption (TDE) for Azure SQL Database	Protects data on the disk Key management is done by Azure, which makes it easier to obtain compliance
In use (end-to-end)	Always Encrypted for client-side column encryption	Data is protected end-to-end, but the application is aware of encrypted columns This is used in the absence of data masking and TDE for compliance-related scenarios



Transparent Data Encryption

All customer data encrypted at rest

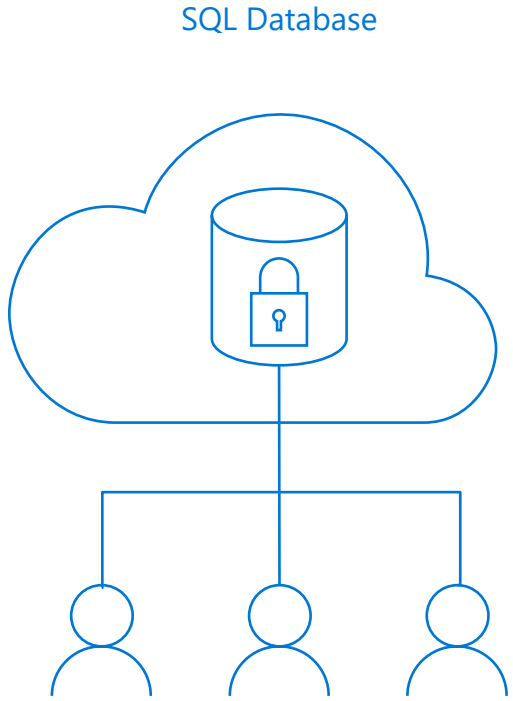
Encryption keys managed by Azure

Application changes kept to a minimum

Transparent encryption/decryption of data in a TCE-enabled client driver

Support for equality operations (including joins) on encrypted data

Bring You Own Key (BYOK) supported



Always Encrypted

Overview

Protect data at rest and in motion, on premises and in the cloud

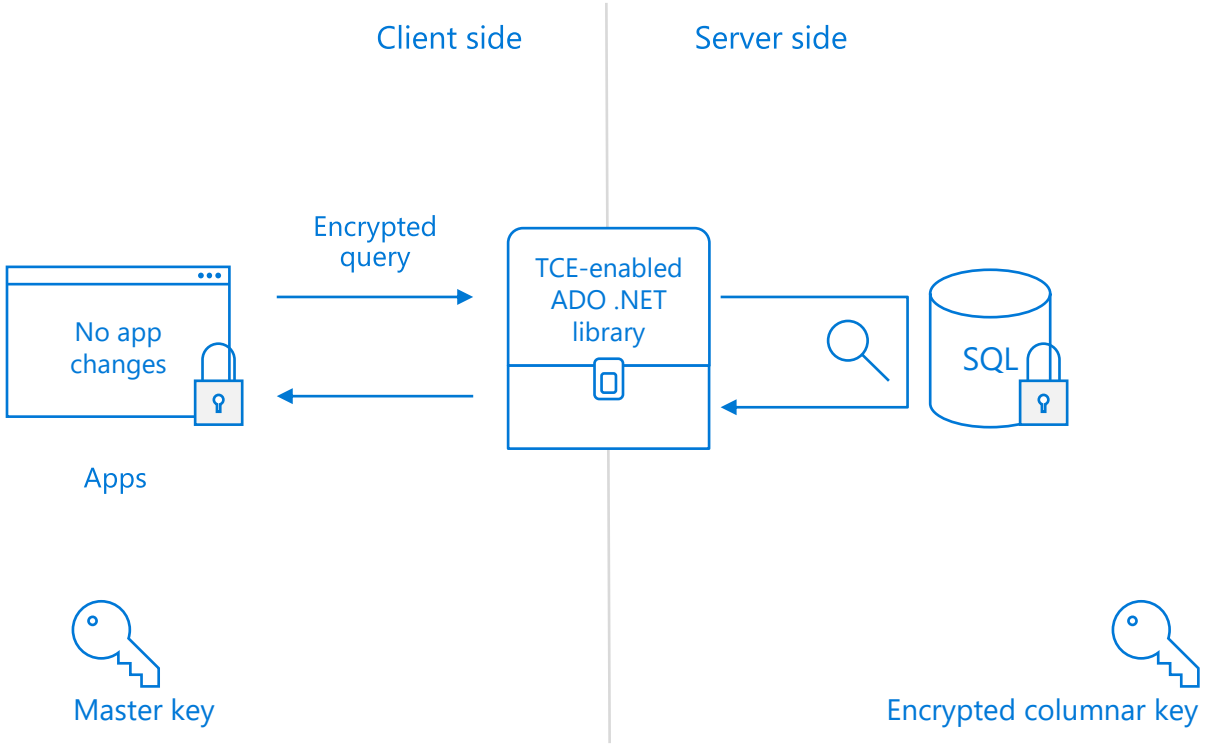
Transparent client-side encryption, while SQL Server executes T-SQL queries on encrypted data

Benefits

Sensitive data remains encrypted and queryable at all times
on-premises and in the cloud

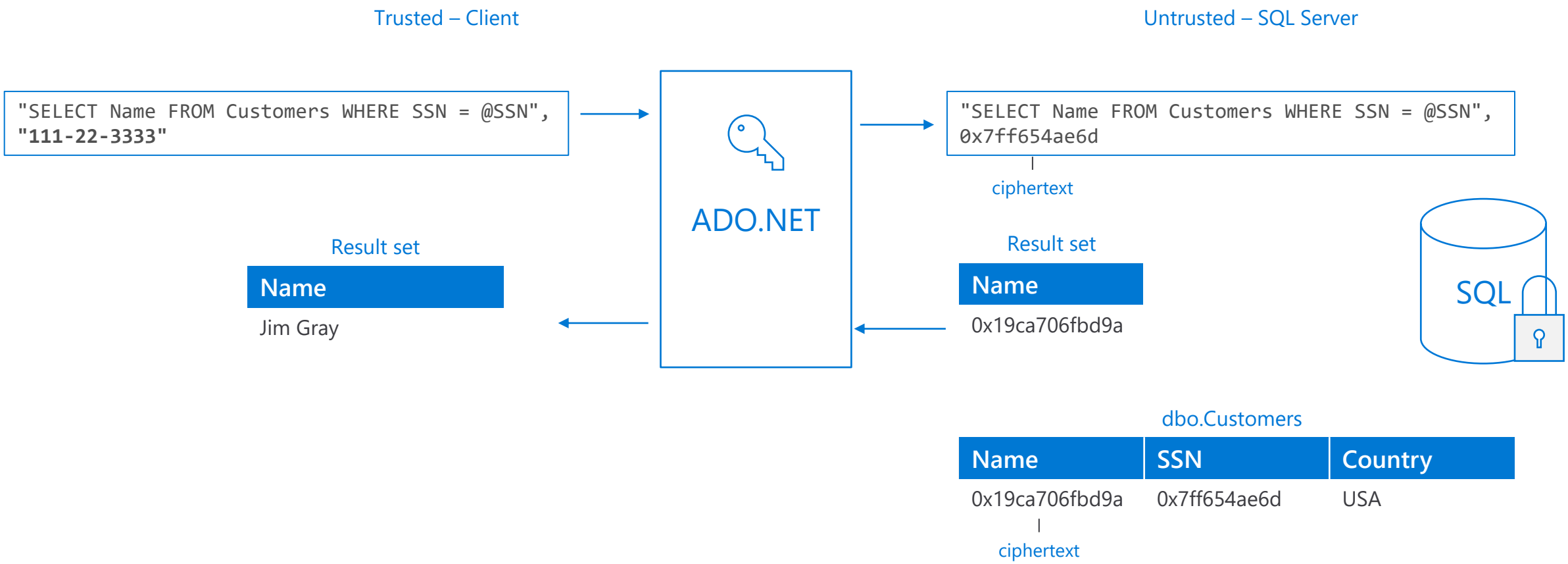
Unauthorized users never have access to data or keys

No application changes



How Always Encrypted works

Encrypted sensitive data and its corresponding keys are never seen in plaintext in SQL Server



Protect data from high-privileged, unauthorized users

Client-side encryption

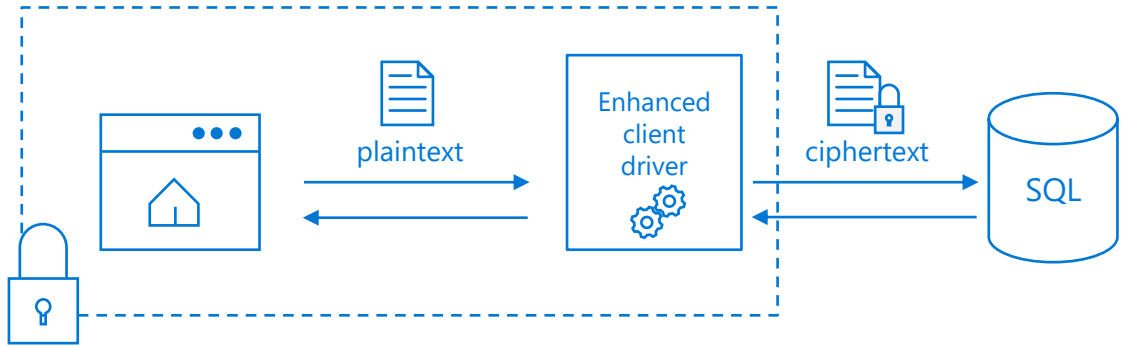
Sensitive data and related encryption keys are never revealed to the database engine

Encryption transparency

Client driver transparently encrypts query parameters and decrypts encrypted results

Queries on encrypted data

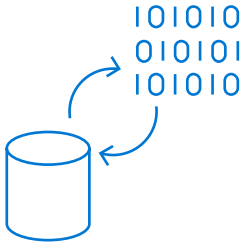
Support for equality comparison on columns encrypted using deterministic encryption



Limitations of Always Encrypted

Reduced functionality of queries on encrypted columns

Encrypted columns only allow equality comparisons



Data needs to be moved out of the database for initial encryption and key rotation

This process can be time consuming and prone to network errors

Confidential computing

Overview

Confidential computing allows data to be protected inside a Trusted Execution Environment (TEE), also known as an enclave

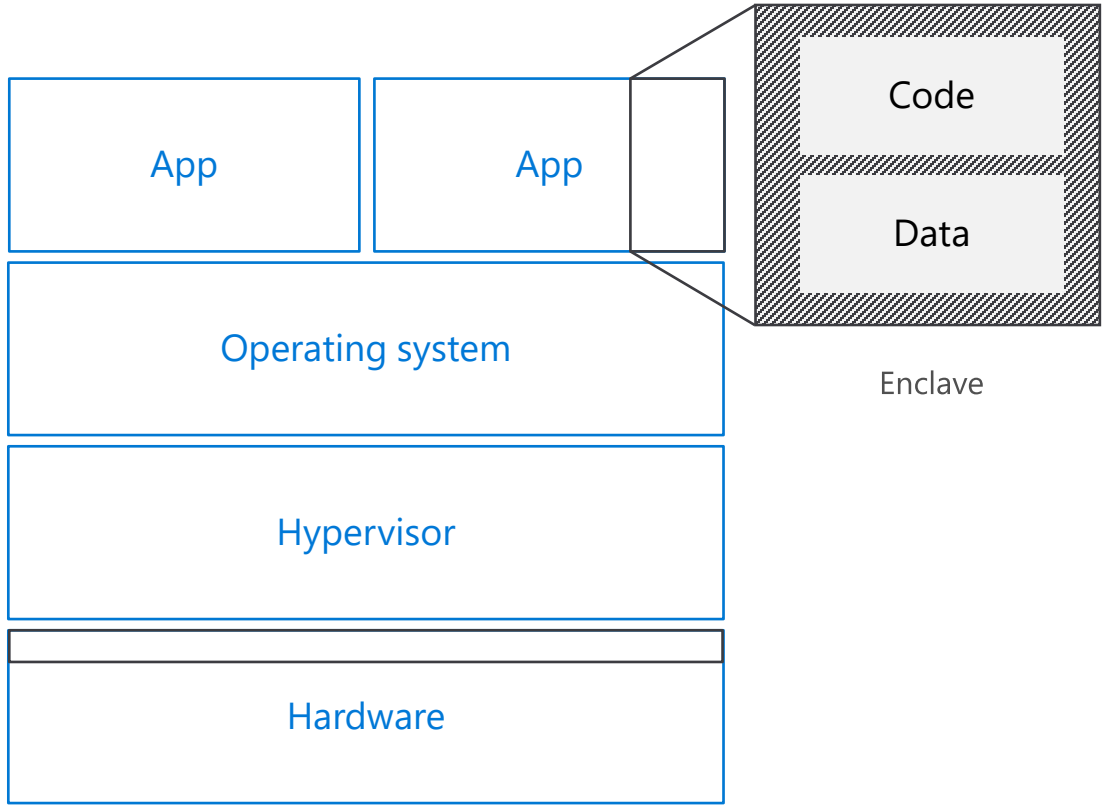
An enclave is a protected region of memory that appears as a black box to the containing process and the OS

Microsoft supports SGX and VSM TEEs

Benefits

Only authorized code is permitted to run inside an enclave

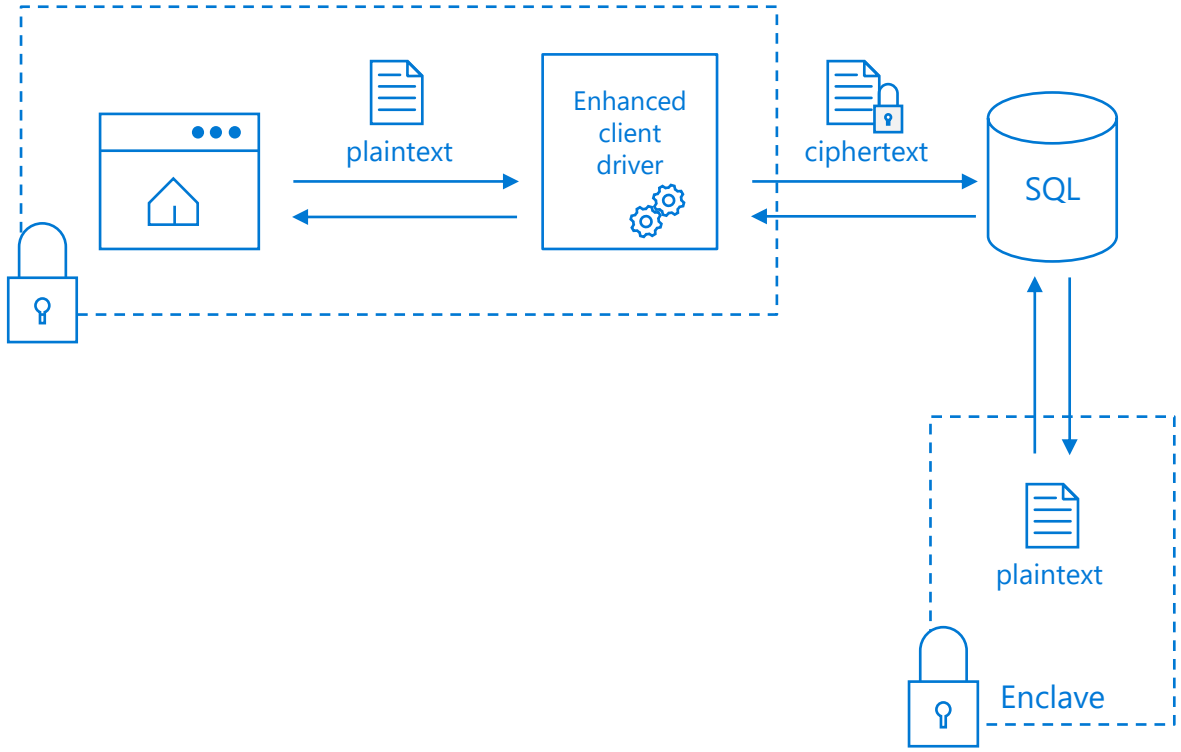
Both the data and the code inside the enclave are inaccessible from the outside and protected from malicious insiders, hackers, and malware



Data protection inside a Trusted Execution Environment

Secure computations inside the enclave

When processing queries, the SQL Server database engine delegates rich computations and cryptographic operations on encrypted columns to the enclave, where the data is safely decrypted and processed



Benefits of Always Encrypted using enclaves

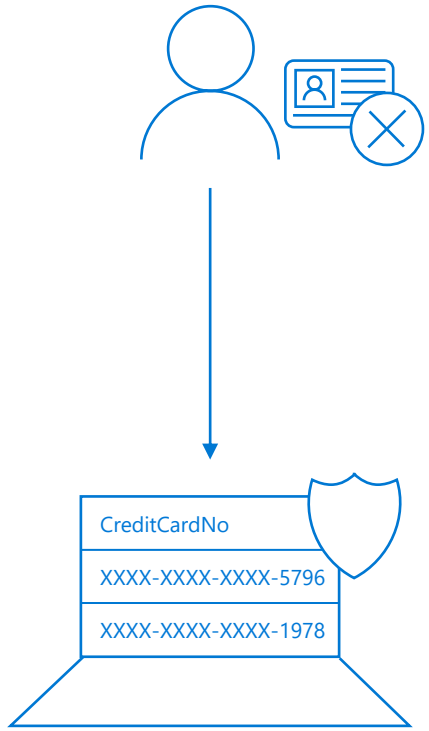
Protect your data in use from high-privilege but unauthorized users, including:

Malicious insiders, hackers and malware, and third-party access without consent

Avoid the pain of moving data

Execute rich computations in client apps without needing to move sensitive data to the client side

Perform initial encryption or key rotation without moving data with in-place encryption



Steps to employing Always Encrypted using enclaves

1. **Configure an enclave-enabled column master key (CMK)**
   ENCLAVE_COMPUTATIONS property is specified in CMK metadata
   The CMK is used to protect column encryption keys (CEK)

2. **Configure a enclave-enabled column-encryption key (CEK)**
   A CEK that is encrypted with an enclave-enabled CMK
   Only enclave-enabled CEKs are permitted to be shared with the enclave

3. **Create enclave-enabled columns**
   Columns encrypted with enclave-enabled CEKs
   Data stored in enclave-enabled columns can be processed inside the enclave

Always Encrypted using enclaves

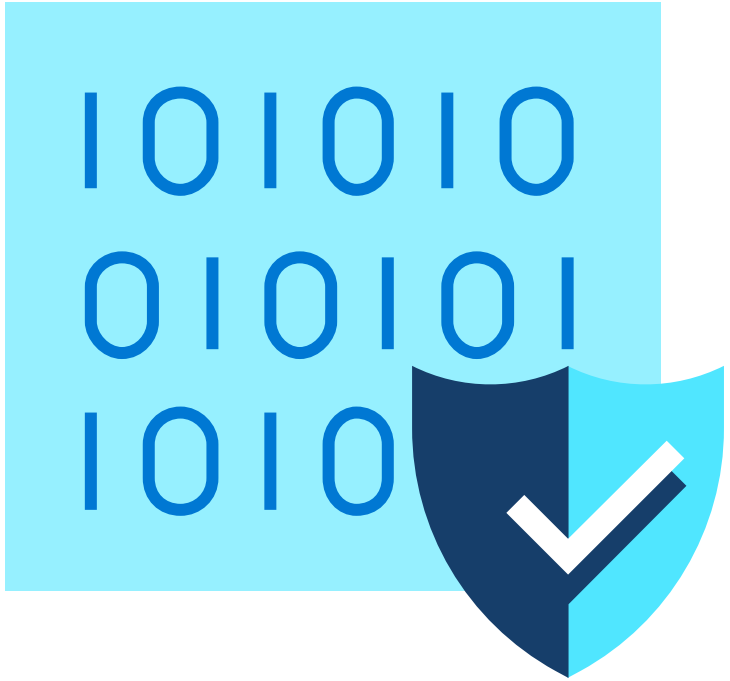
A new era in cybersecurity—confidential computing brings secure enclaves to Azure

Trusted execution environments protect data in use

First cloud to offer Intel Software Guard Extensions (SGX) enclaves

Sign up for an early access preview

https://aka.ms/SQLEnclavesPreview



Secure state best practices

Security updates

Keep up with recommended settings

Visibility

Track changes to identify vulnerabilities

Compliance

Perform regulatory audits and internal or external reviews

Vulnerability Assessment

Get visibility

Discover sensitive data and potential security holes

Remediate

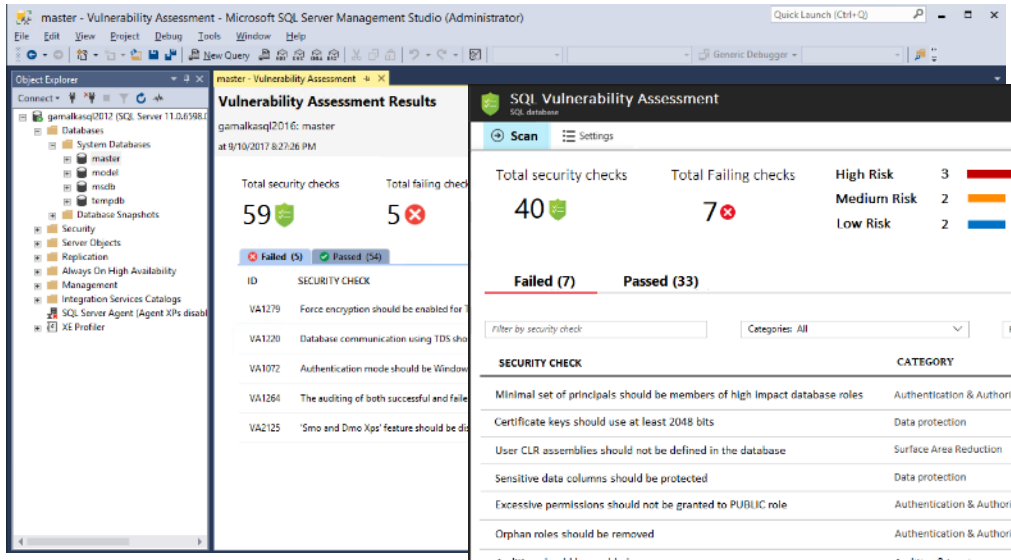
Actionable remediation and security hardening steps

Customize

Baseline policy tuned to your environment, allowing you to focus on deviations

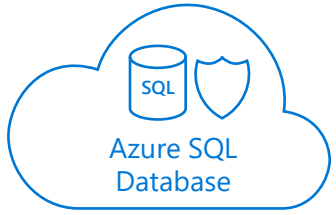
Report

Pass internal or external audits to facilitate compliance



SQL Server on-premises

Azure SQL Database

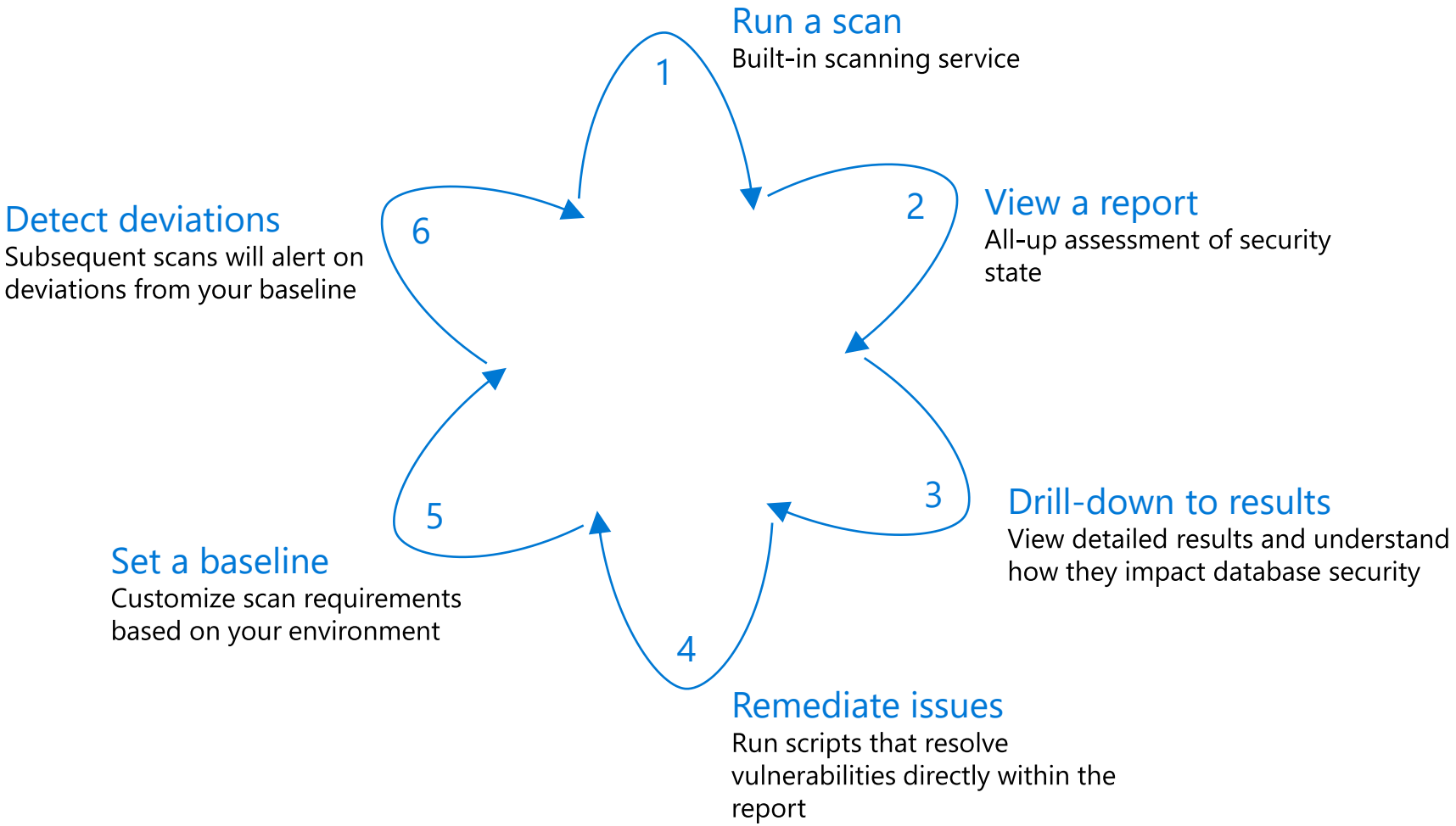


Vulnerability Assessment

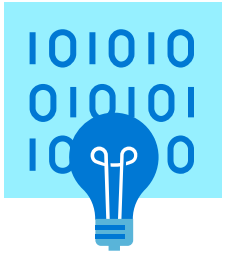
Identifies, tracks, and resolves SQL security vulnerabilities

Developer/DBA

Using Vulnerability Assessment



Information Protection



Discover

Will be integrated into the Microsoft Information Protection framework
Protects data and moves across database boundaries in your organization

Classify

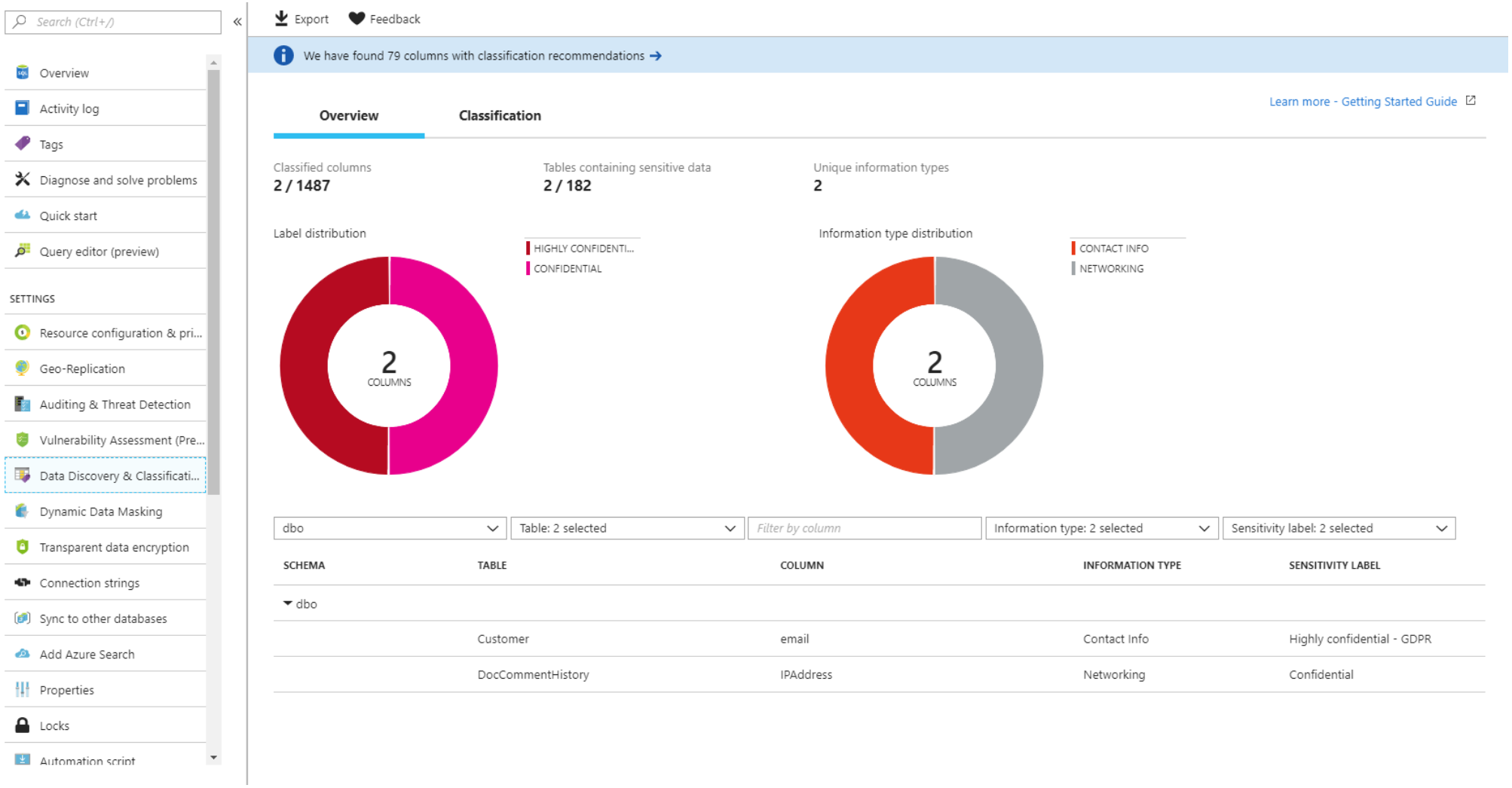
Automatic Data Discovery and recommendations to classify sensitive data
Each column associated with a sensitivity label and label type
Manual classification on data can also be added

Column label persisted as column metadata as new classification attributes in SQL Engine
Export classification information to Excel report for internal or external auditing purposes

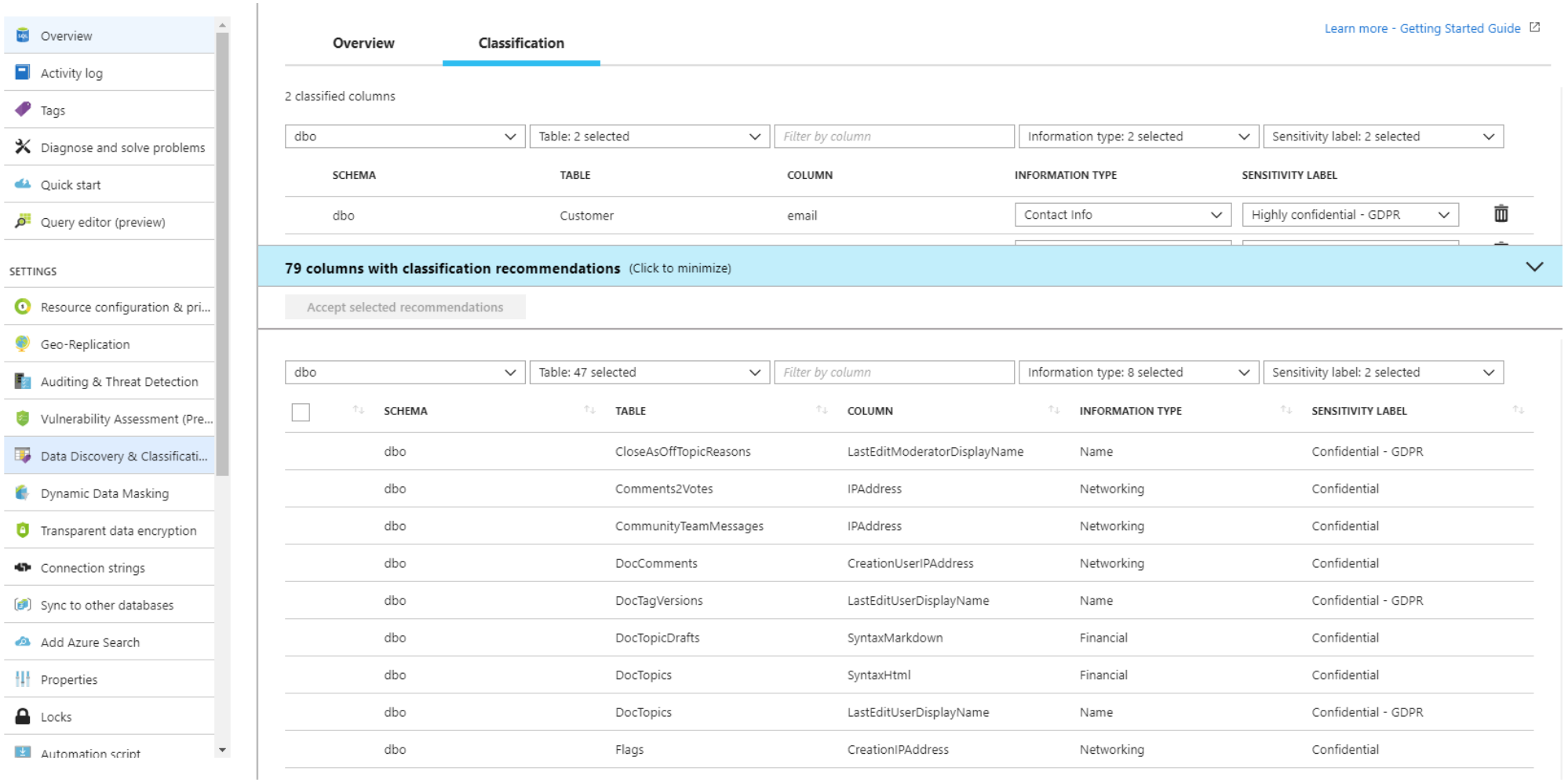
Label

Once a column is labeled it can be used for auditing and protection purposes
All labeled columns will be fully audited at run time for any queries that access them
Auditing will monitor which users access sensitive data and how much sensitive data they access

Data discovery & classification



Data discovery & Classification



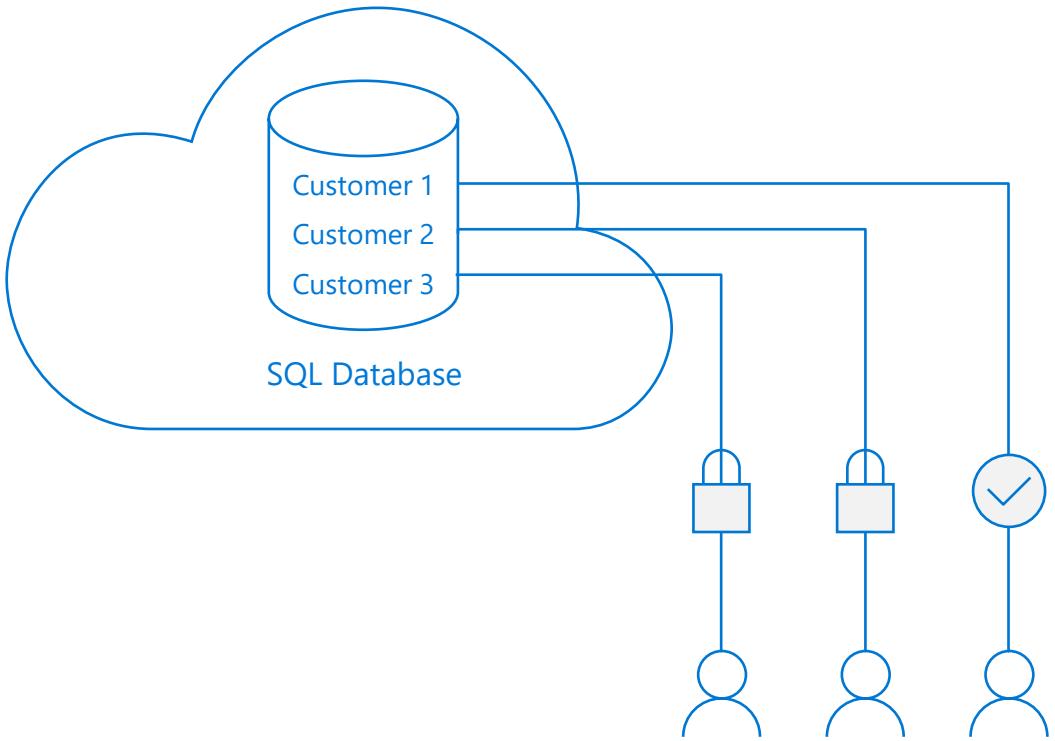
Row-level security

Control access of specific rows in a database table

Help prevent unauthorized access when multiple users share the same tables, or implement connection filtering in multitenant applications

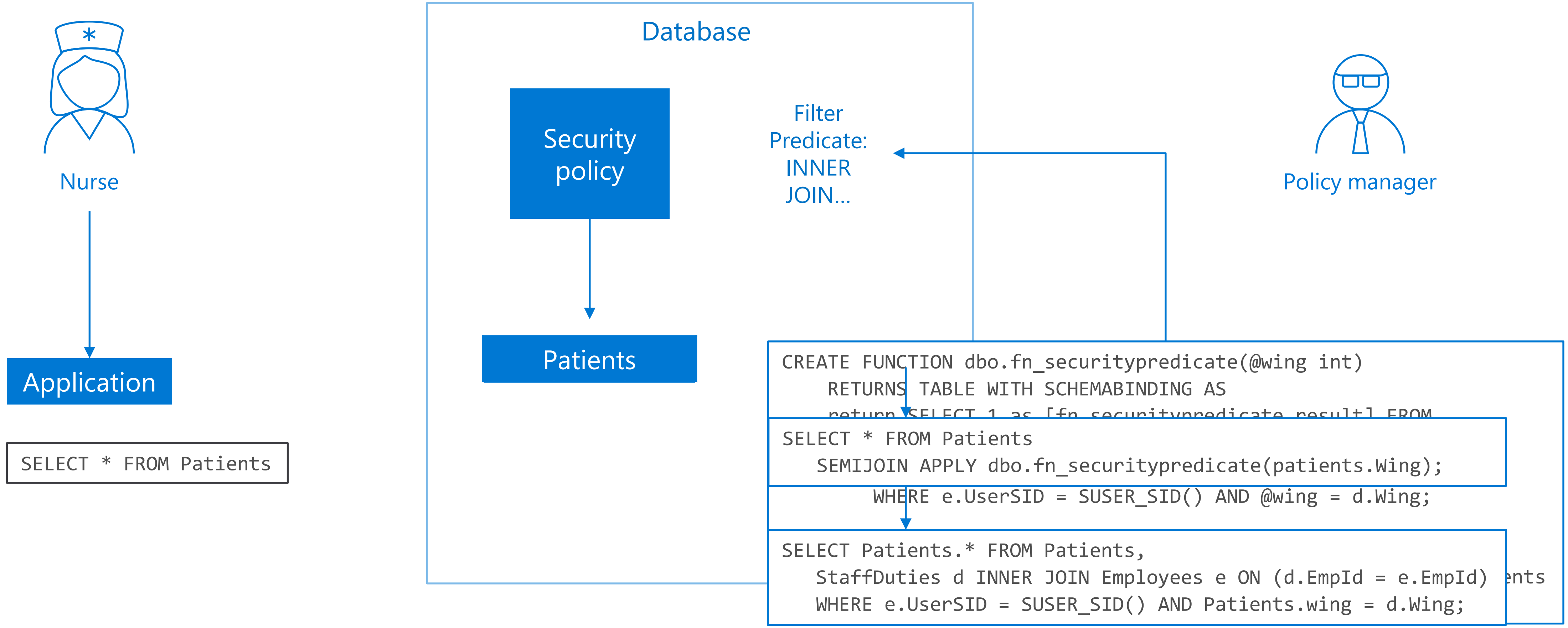
Administer via SQL Server Management Studio or SQL Server Data Tools

Easily locate enforcement logic inside the database and schema bound to the table



RLS in three steps

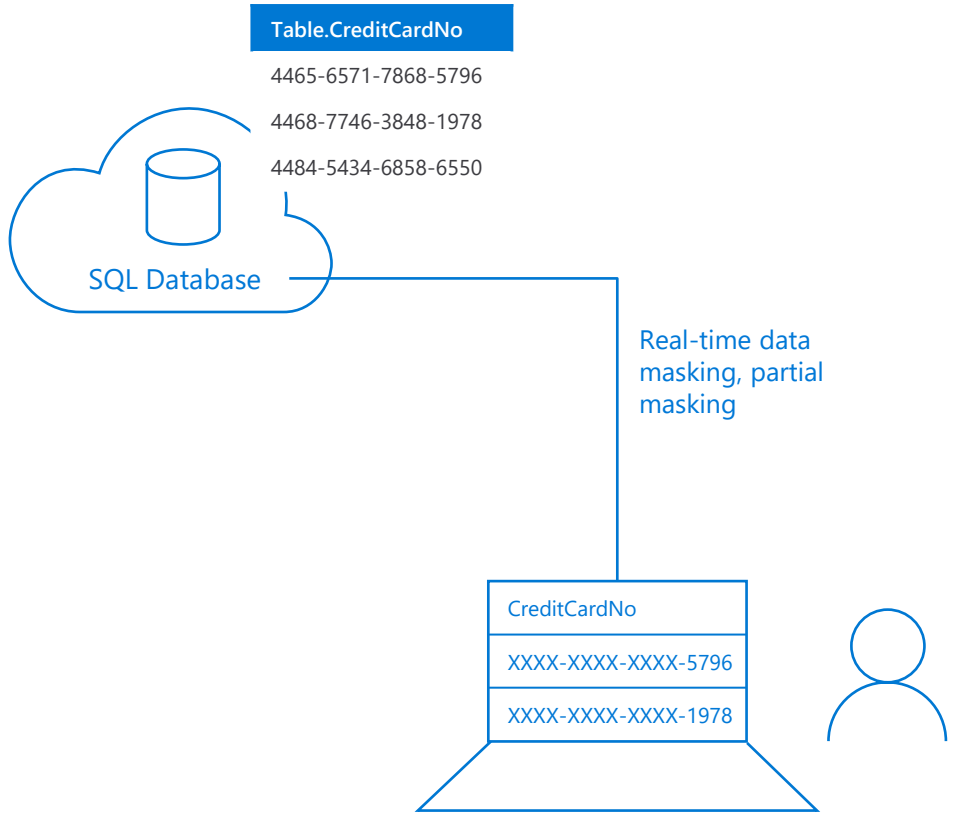
... binding the predicate to the Patients table



Dynamic data masking

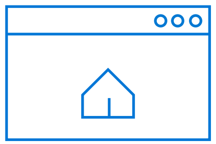
Limit the exposure of sensitive data by hiding it from users

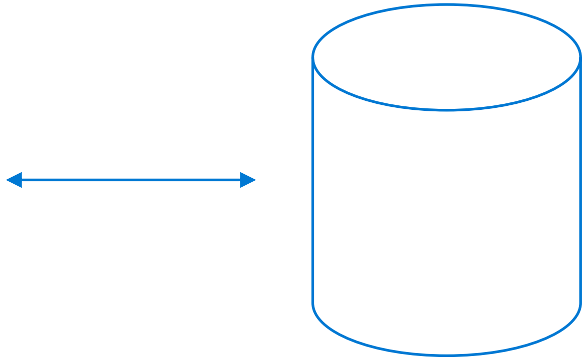
- ✓ **Auto-discovery of potentially sensitive data to mask**
- ✓ **Configurable masking policy from Azure Portal or via DDL in the Server**
- ✓ **On-the-fly obfuscation of data in query results**
- ✓ **Flexibility to define a set of privileged SQL users for un-masked data access**



Dynamic data masking

1. Security officer defines dynamic data masking policy in T-SQL over sensitive data in the Employee table





```
ALTER TABLE [Employee] ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'SSN()'

ALTER TABLE [Employee] ALTER COLUMN [Email]
ADD MASKED WITH (FUNCTION = 'EMAIL()')

ALTER TABLE [Employee] ALTER COLUMN [Salary]
ADD MASKED WITH (FUNCTION = 'RANDOM(1,20000)')

GRANT UNMASK to admin1
```


```
SELECT [Name],
       [SocialSecurityNumber],
       [Email],
       [Salary]
FROM [Employee]
```

other logon

	First Name	Social Security Number	Email	Salary
1	LILA	XXX-XX-XX37	lXX@XXXX.net	8940
2	JAMIE	XXX-XX-XX14	jXX@XXXX.com	19582
3	SHELLEY	XXX-XX-XX28	sXX@XXXX.net	3713
4	MARCELLA	XXX-XX-XX65	mXX@XXXX.net	11572
5	GILBERT	XXX-XX-XX87	gXX@XXXX.net	4487

admin1 logon

	First Name	Social Security Num...	Email	Salary
1	LILA	758-10-9637	lila.barnett@comcast.net	1012794
2	JAMIE	113-29-4314	jamie.brown@ntlworld.com	1025713
3	SHELLEY	550-72-2028	shelley.lynn@charter.net	1040131
4	MARCELLA	903-94-5665	marcella.estrada@comcast.net	1040753
5	GILBERT	376-79-4787	gilbert.juarez@verizon.net	1041308

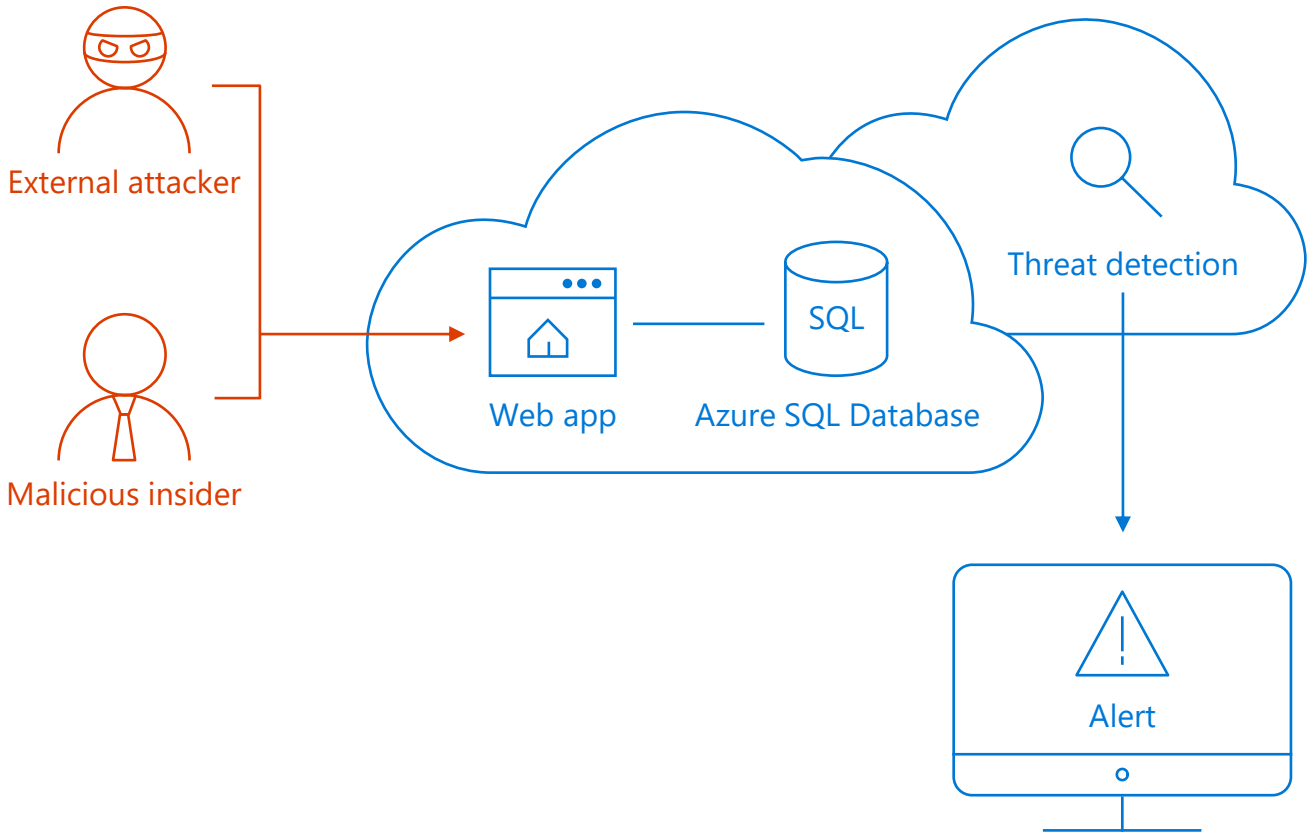
Threat detection

Detect anomalous database activities that could indicate a potential threat

Configure threat detection policy in Azure Portal

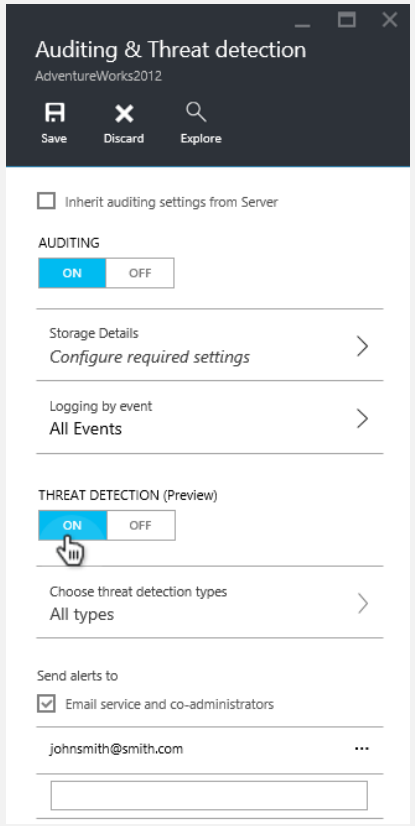
Receive alerts from multiple database threat detectors that identify anomalous activities

Explore audit log around the time of an event

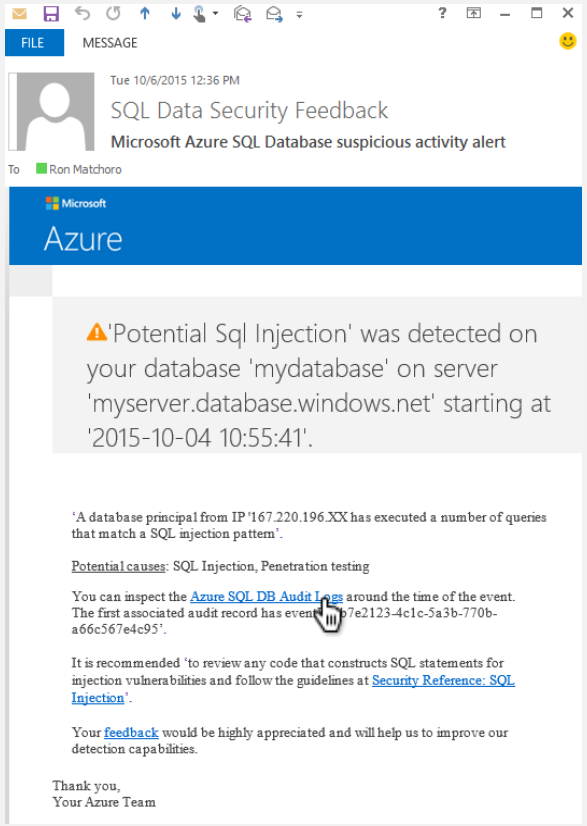


How threat detection works

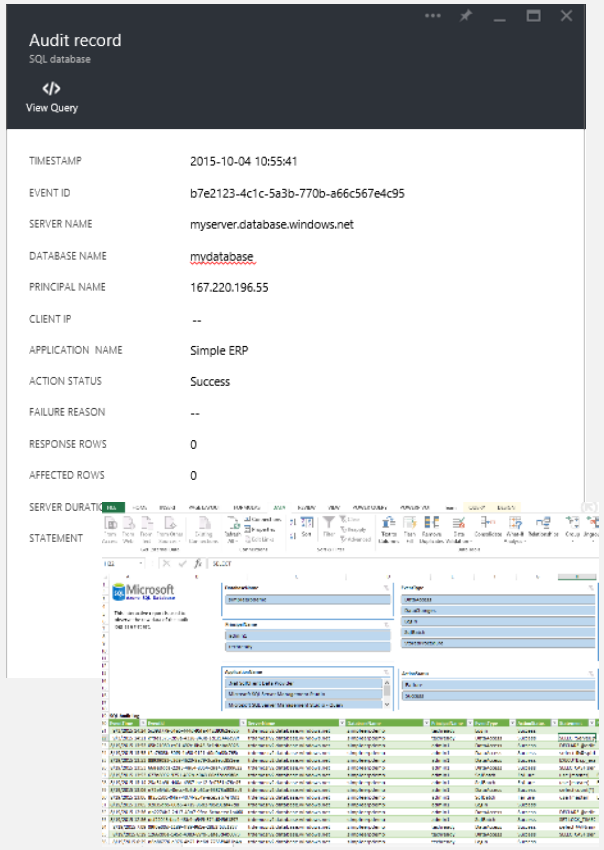
Set up



Alert



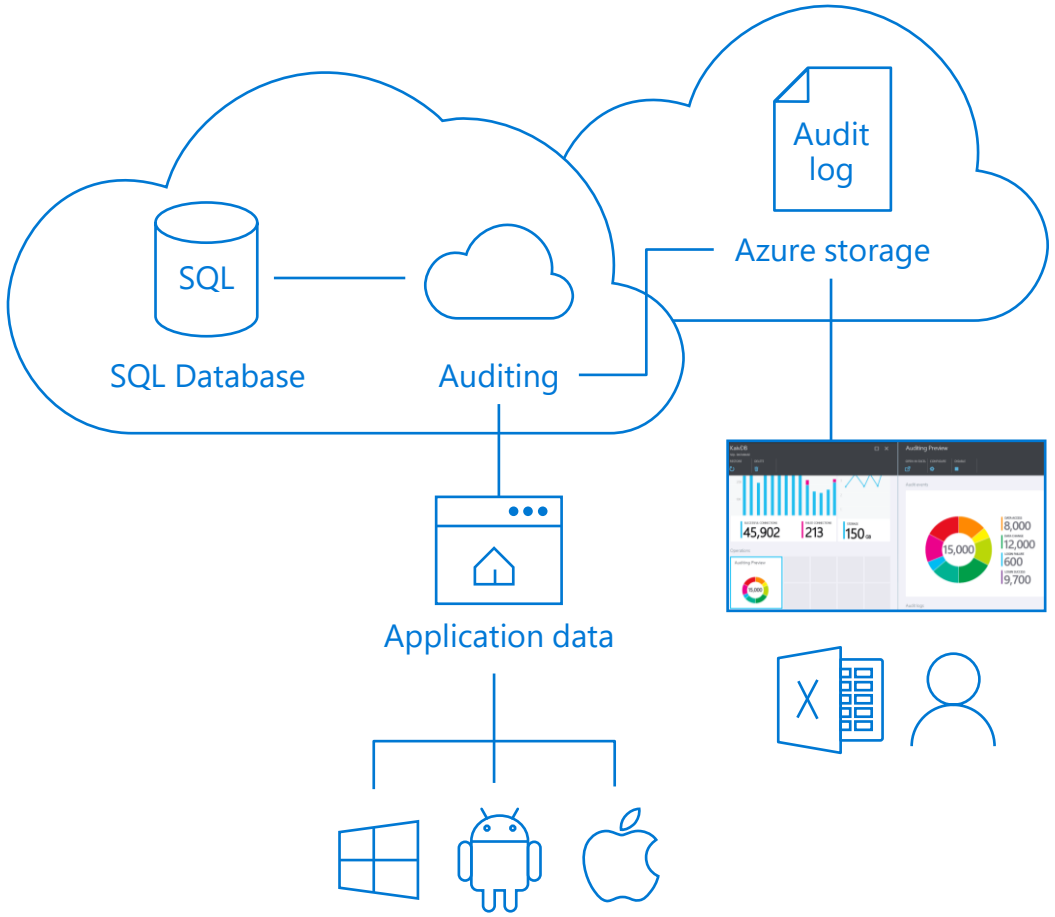
Explore



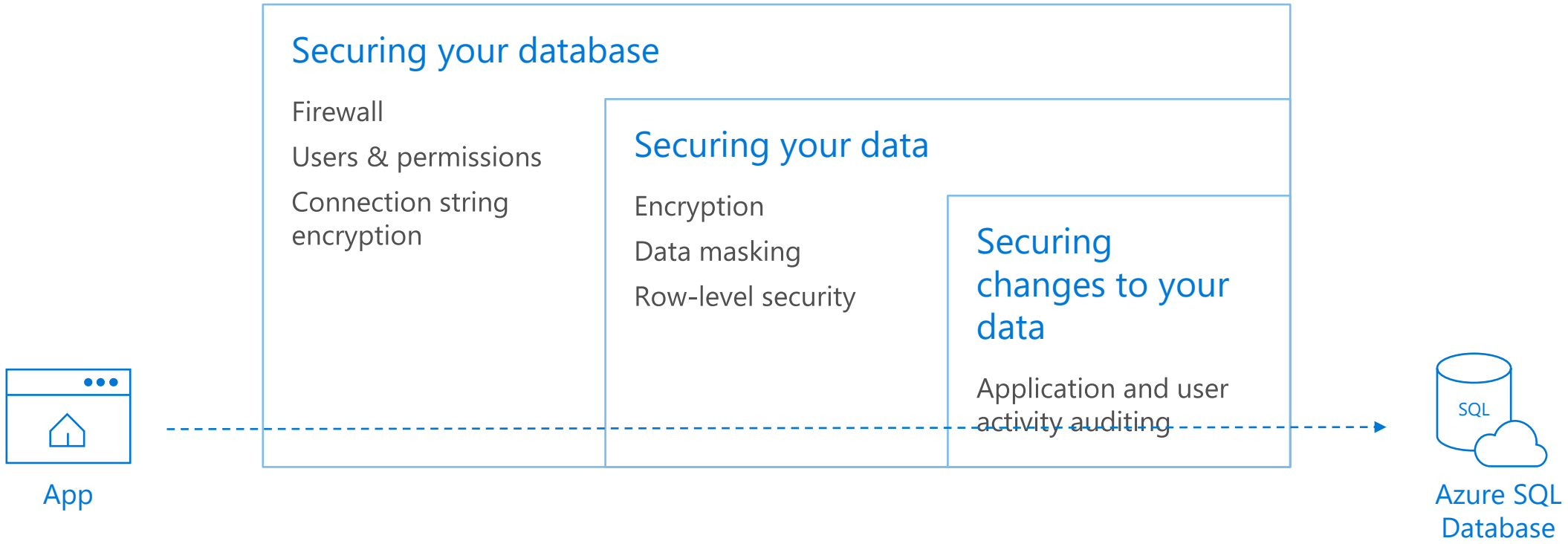
Azure SQL Database Auditing

Gain insight into database events and streamline compliance-related tasks

- ✓ Configurable audit policy via the Azure portal and standard API
- ✓ Audit logs reside in your Azure Storage account, or can be sent directly to Log Analytics or Event Hub
- ✓ Azure portal viewer and SSMS for analysis of audit log
- ✓ Compatible with SQL Server box auditing, including high granularity in defining audit policy



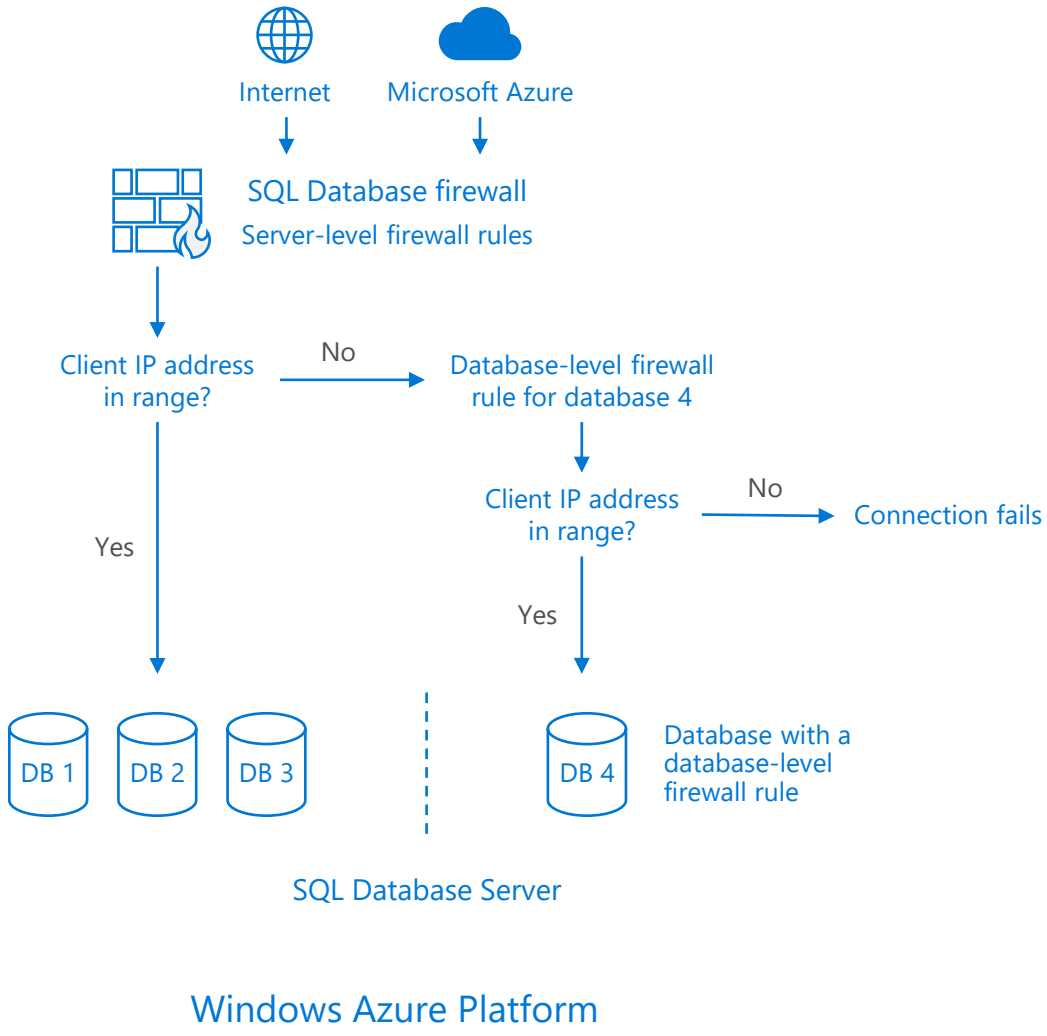
Layered approach to security



Securing your database with firewalls

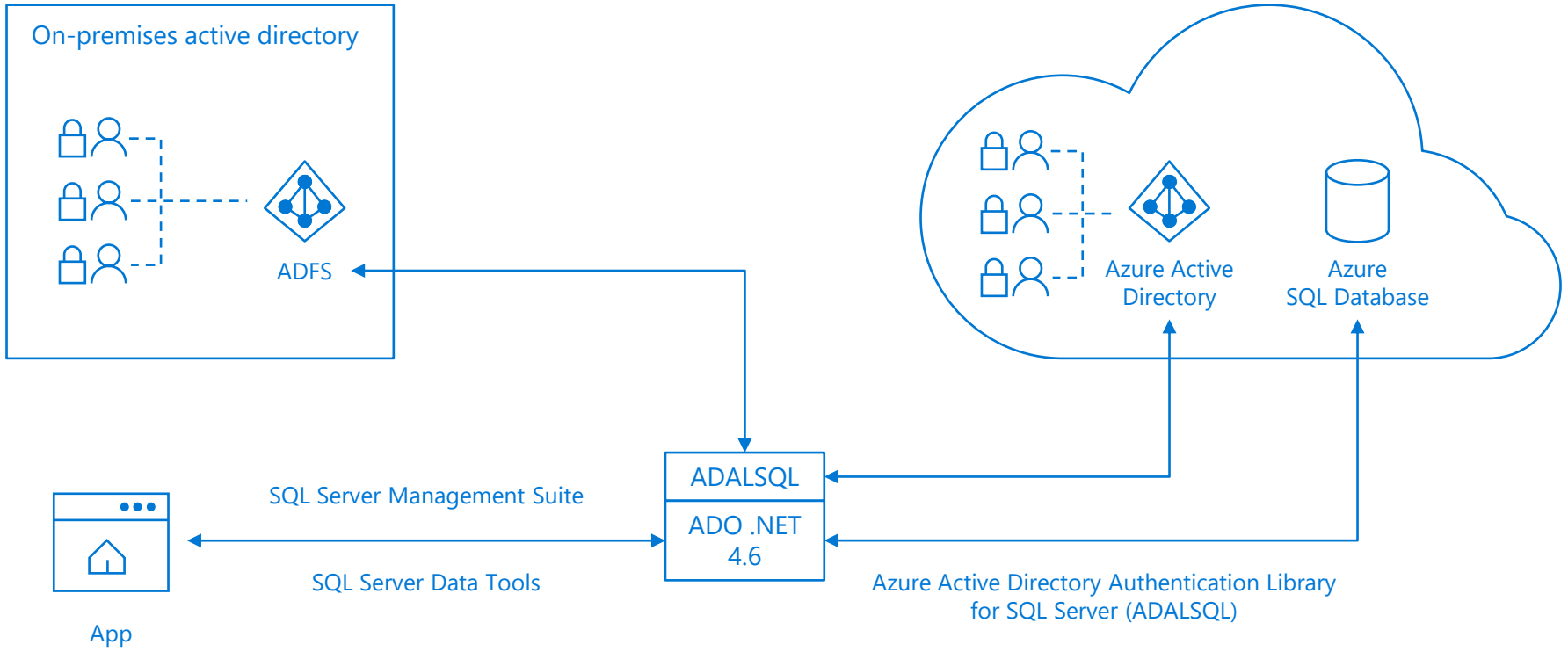
Initially, all access to your Azure SQL Database server is blocked by the firewall

In order to begin using your Azure SQL Database server, you must go to the Management Portal



Trust architecture

Azure Active Directory and Azure SQL Database



Create a security policy

Create a security policy for row-level security

The following examples demonstrate the use of the CREATE SECURITY POLICY syntax

For an example of a complete security policy scenario, see Row-Level Security

```
-- The following syntax creates a security policy with a filter
predicate for the Customer table
CREATE SECURITY POLICY [FederatedSecurityPolicy]
           ADD FILTER PREDICATE
[rls].[fn_securitypredicate]([CustomerId])
           ON [dbo].[Customer];

-- Create a new schema and predicate function, which will use the
application user ID stored in CONTEXT_INFO to filter rows.
CREATE FUNCTION rls.fn_securitypredicate (@AppUserId int)
           RETURNS TABLE
           WITH SCHEMABINDING
AS
RETURN (
SELECT 1 AS fn_securitypredicate_result
WHERE
           DATABASE_PRINCIPAL_ID() = DATABASE_PRINCIPAL_ID('dbo') -
- application context
           AND CONTEXT_INFO() = CONVERT(VARBINARY(128),
@AppUserId));
GO
```

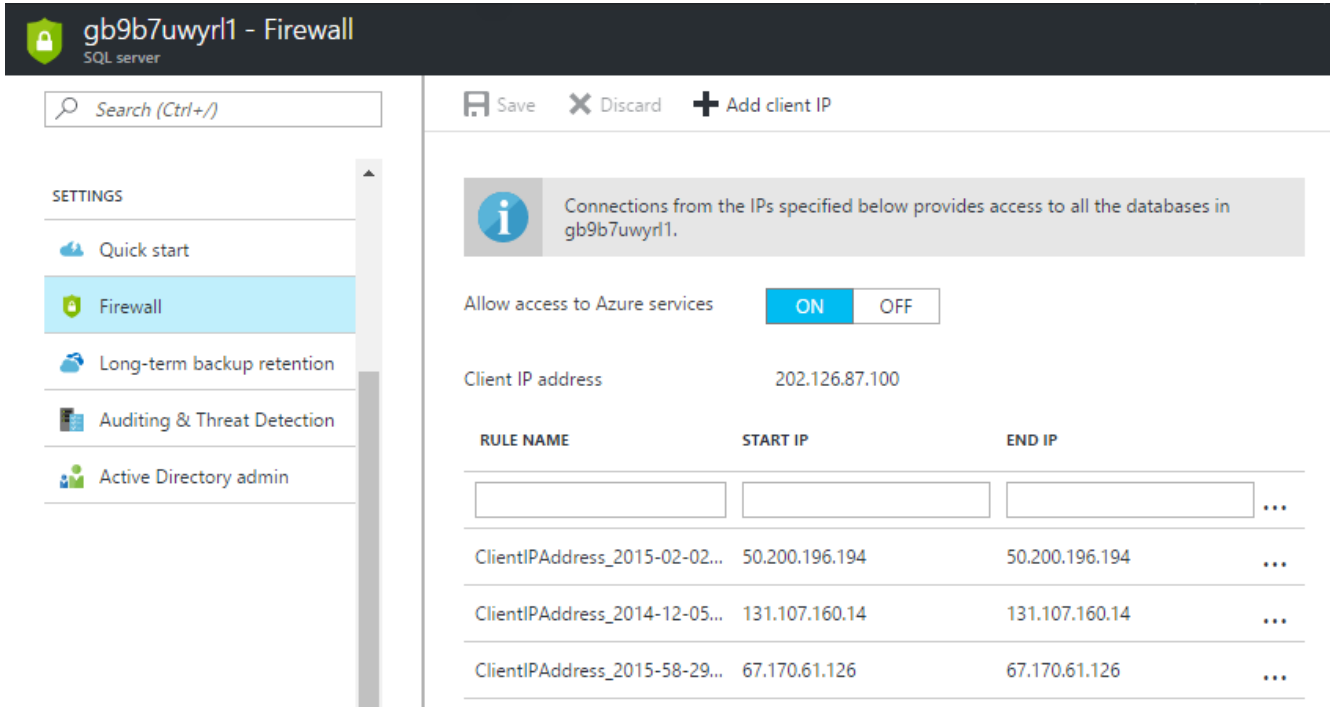
Firewall configuration using portals

By default, Azure blocks all external connections to port 1433

Enable in the following ways in Azure portal:

Classic portal: Server level – Configure page

New portal: Settings > Firewall > Firewall settings blade



Firewall configuration using PowerShell/T-SQL

Manage SQL Database firewall rules using code

Windows PowerShell Azure cmdlets

New-AzureSqlDatabaseServerFirewallRule

Get-AzureSqlDatabaseServerFirewallRule

Set-AzureSqlDatabaseServerFirewallRule

Transact SQL

sp_set_firewall_rule

sp_set_database_firewall_rule

sp_delete_firewall_rule

sp_delete_database_firewall_rule

```
# PS Enable Azure connections
PS C:\> New-AzureSqlDatabaseServerFirewallRule `
          -ServerName "Contoso" `
          -AllowAllAzureServices `
          -RuleName "myRule2"

-- PS Allow external IP access to SQL Database
PS C:\> New-AzureSqlDatabaseServerFirewallRule `
          -ServerName "Contoso" `
          -RuleName "myRule1" `
          -StartIpAddress 12.1.1.1 `
          -EndIpAddress 12.1.1.2

-- T-SQL Enable Azure connections
sp_set_firewall_rule N'Allow Windows Azure',
          '0.0.0.0','0.0.0.0'

-- T-SQL Allow external IP access to SQL Database
sp_set_firewall_rule N'myRule1',
          '12.1.1.1','12.1.1.2'
```

Firewall configuration using REST API

Managing firewall rules through REST API must be authenticated

For information, see Authenticating Service Management Requests

Server-level rules can be created, updated, or deleted using REST API

To create or update a server-level firewall rule, execute the POST method

To remove an existing server-level firewall rule, execute the DELETE method

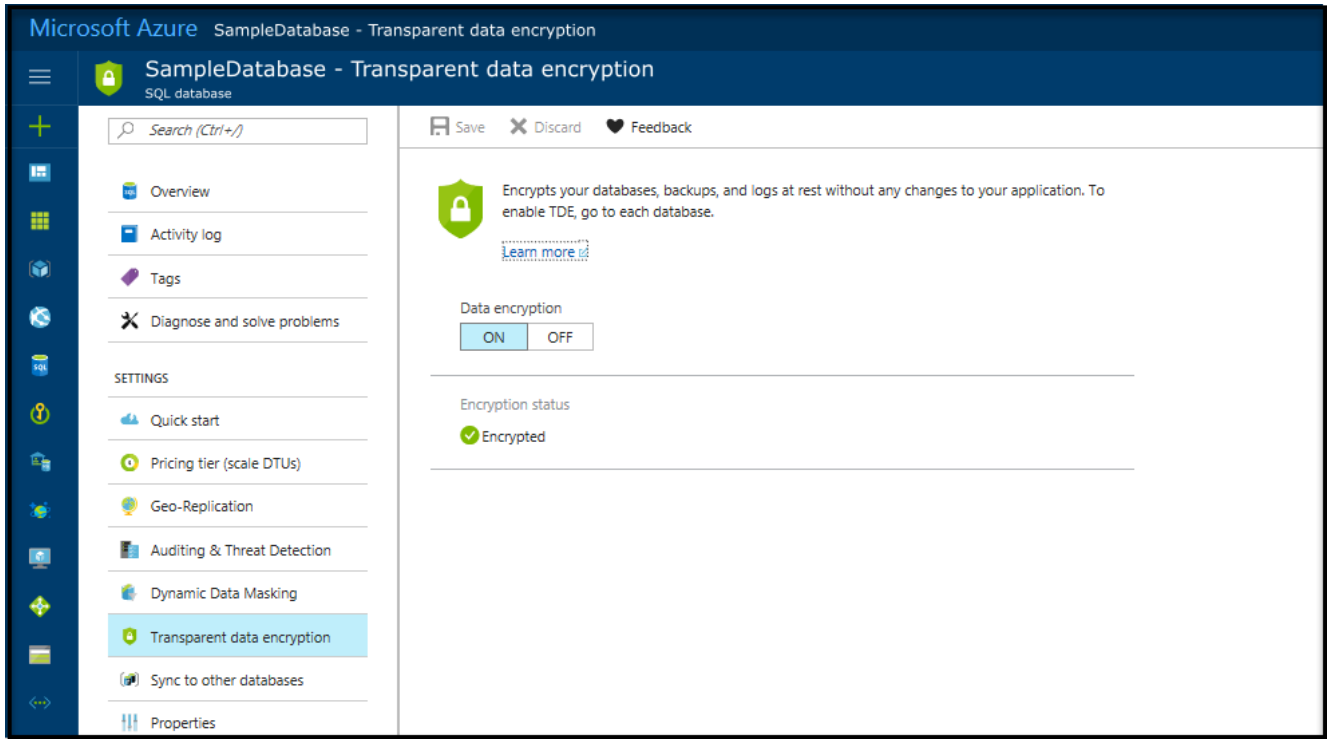
To list firewall rules, execute the GET

```
POST
https://management.core.windows.net:8443/{subscriptionId}
/services/sqlservers/servers/Contoso/firewallrules
REQUEST BODY
<ServiceResource
xmlns="http://schemas.microsoft.com/windowsazure">
<Name>myRule1</Name>
          <StartIPAddress> 12.1.1.1 </StartIPAddress>
          <EndIPAddress> 12.1.1.1 </EndIPAddress>
</ServiceResource>

DELETE
https://management.core.windows.net:8443/{subscriptionId}
/services/sqlservers/servers/Contoso/firewallrules/myRule
1

GET
https://management.core.windows.net:8443/{subscriptionId}
/services/sqlservers/servers/Contoso/firewallrules
```

Customer experience

Customer can verify that the database is encrypted in portal or via PowerShell/T-SQL

```
Get-AzSqlDatabaseTransparentDataEncryption
[-ServerName] <String>
[-DatabaseName] <String>
[-ResourceGroupName] <String>
[-DefaultProfile <IAzureContextContainer>]
[-WhatIf]
[-Confirm] [<CommonParameters>]
```




```
Get-AzSqlDatabaseTransparentDataEncryption -ServerName "server01" -ResourceGroupName "resourcegroup01" -DatabaseName "database01"

ResourceGroupName       ServerName      DatabaseName        State
-----------------       ----------      ------------        -----
resourcegroup01         server01        database01          Disabled
```

Minimal impact on applications

Existing application queries will not be affected

Sensitive data masked in the result set of a query

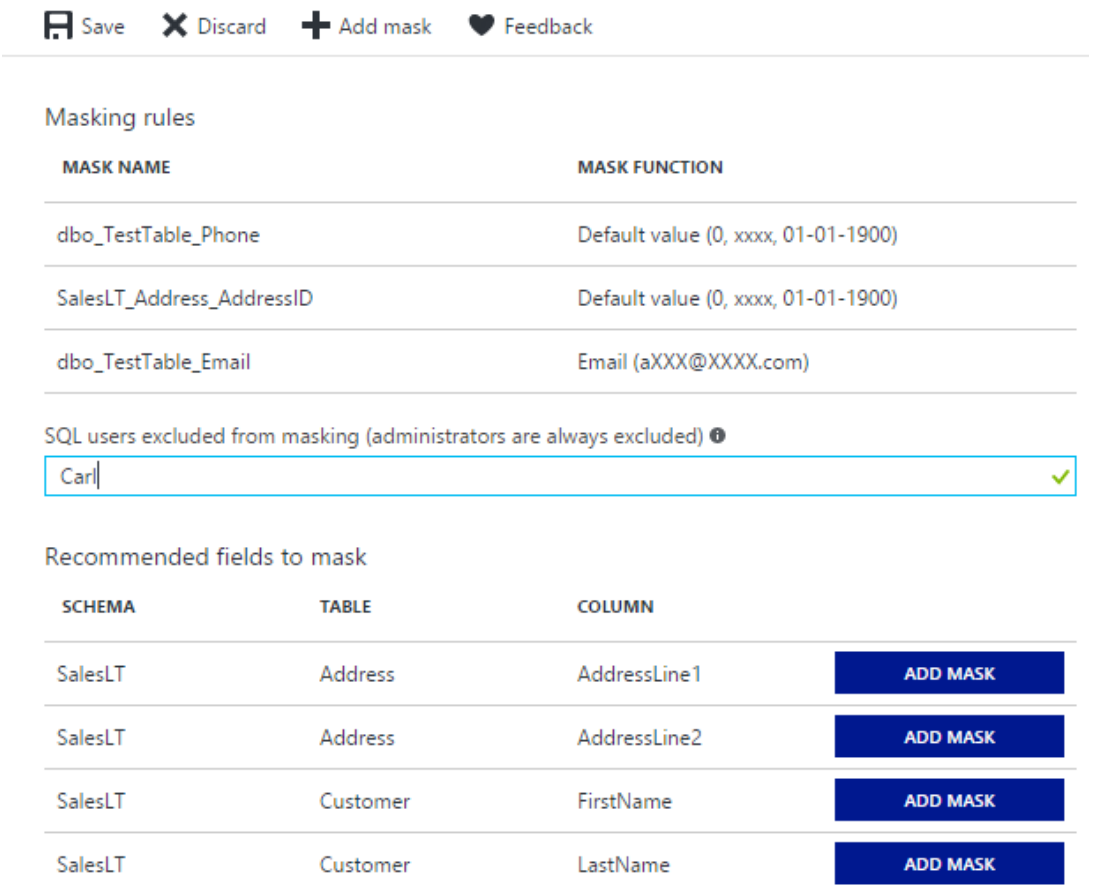
Option to restrict masking policy is available to exclude specific developers

Complementary to other security features that protect sensitive data

Limit access by setting firewall rules and user roles with minimal permissions

Encrypt sensitive data at rest

Use auditing for tracking discrepancies and anomalous activities



Security administration overview

Point of difference	On-premises SQL Server	Microsoft Azure SQL Database
Where you manage server-level security	The Security folder in SQL Server Management Studio's Object Explorer	The master database
Server-level security role for creating logins	`securityadmin` fixed server role For more information, see Server-Level Roles	`loginmanager` database role in the master database

Commands for managing logins	`CREATE LOGIN`
	`ALTER LOGIN`
	`DROP LOGIN`
		`CREATE LOGIN`
		`ALTER LOGIN`
		`DROP LOGIN`
		(There are some parameter limitations and you must be connected to the master database)

View that shows all logins	`sys.syslogins` (`sys.sql_logins` for SQL Server authentication logins)	`sys.sql_logins` (You must be connected to the master database)
Server-level role for creating databases	`dbcreator` fixed database role For more information, see Server-Level Roles	`dbmanager` database role in the master database
Command for creating a database	`CREATE DATABASE`	`CREATE DATABASE` (There are some parameter limitations and you must be connected to the master database)
Dropping databases	`DROP DATABASE`	`DROP DATABASE` If a user is in the dbmanager role, they have permission to `DROP` any database, regardless of which user originally created it.
View that lists all databases	`sys.databases` (view)	`sys.databases` (You must be connected to the master database)

The largest compliance portfolio in the industry

ISO 27001

SOC 1 Type 2

SOC 2 Type 2

PCI DSS Level 1

Cloud Controls Matrix

ISO 27018

Content Delivery and Security Association

Shared Assessments

FedRAMP JAB P-ATO

HIPAA / HITECH

FIPS 140-2

21 CFR Part 11

FERPA

DISA Level 2

CJIS

IRS 1075

ITAR-ready

Section 508 VPAT

European Union Model Clauses

EU Safe Harbor

United Kingdom G-Cloud

China Multi Layer Protection Scheme

China GB 18030

China CCCPPF

Singapore MTCS Level 3

Australian Signals Directorate

New Zealand GCIO

Japan Financial Services

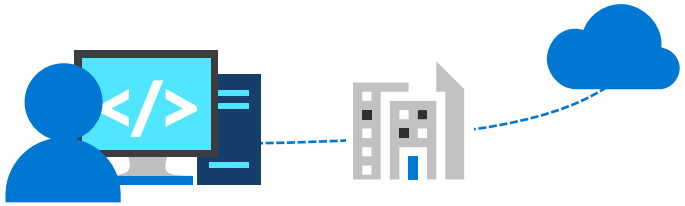
ENISA IAF

SaaS patterns and multi-tenancy data models

Common app scenarios

Enterprise apps serving customers

Create new or re-architect apps to meet customer needs to augment their products and services



Cloud-born SaaS

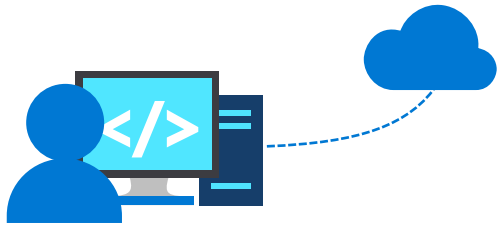
Disrupting traditional apps and business models

Common app types

Mobile/social
e-Commerce
Accounting
Payroll
Global marketing sites
Catalog
Gaming/voting
...and many others

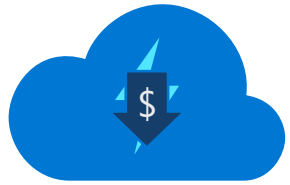
Enterprise apps serving employees

New or re-architected apps for internal use, reducing CAPEX and OPEX

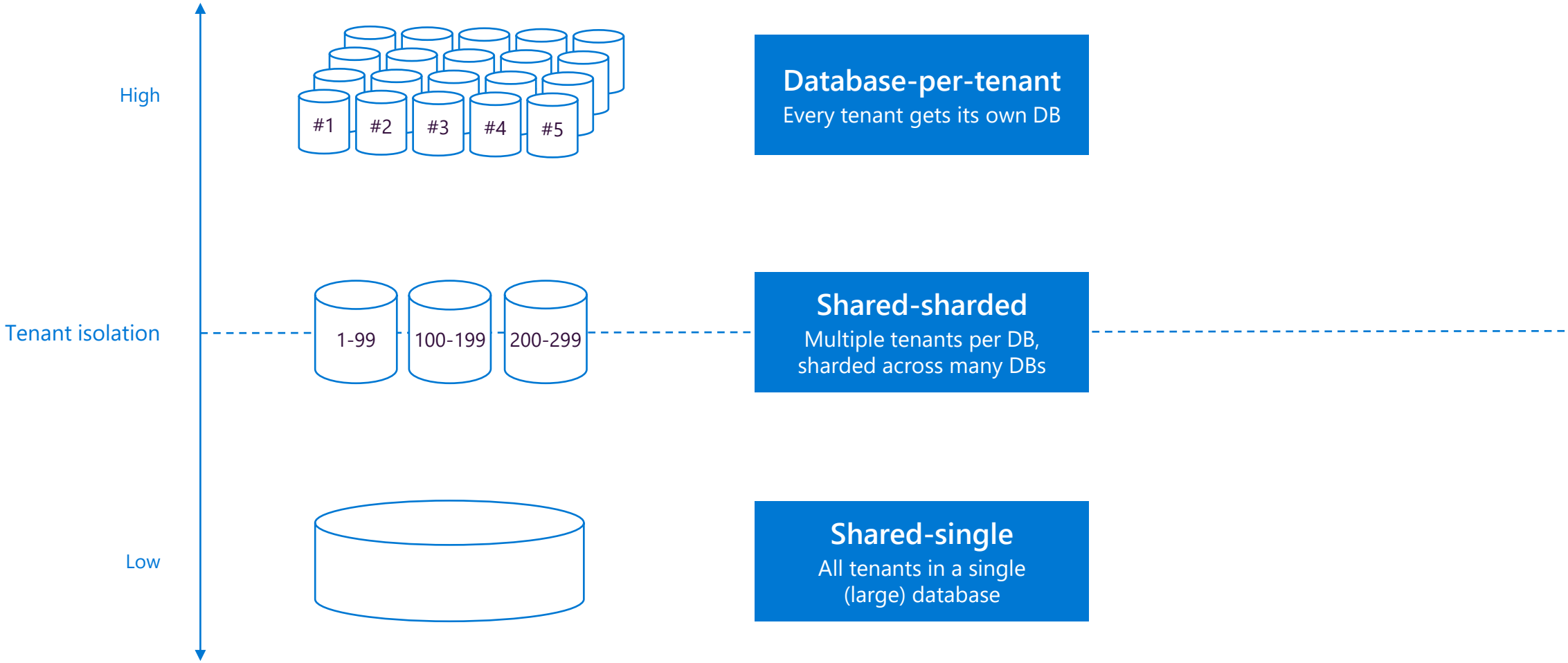


ISVs going SaaS

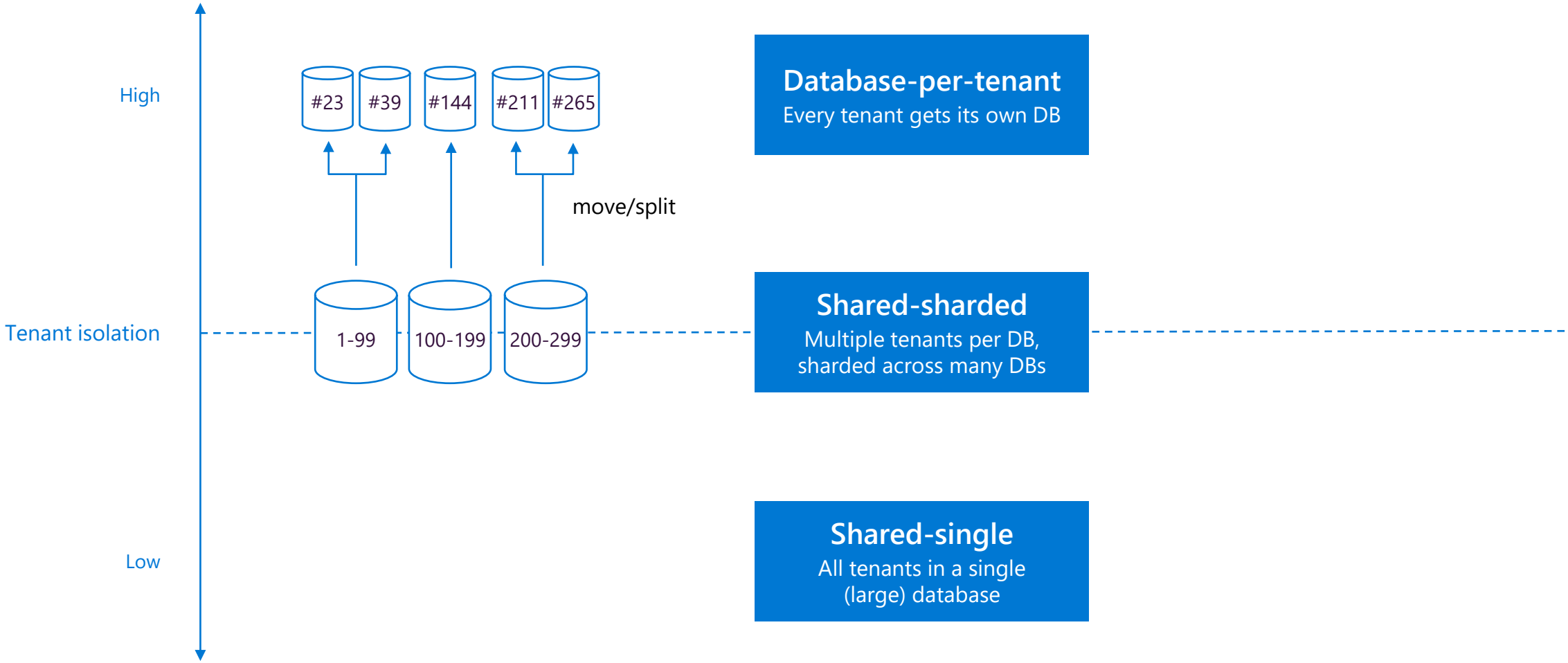
Re-architect applications to transition from selling software licenses to subscription-based SaaS



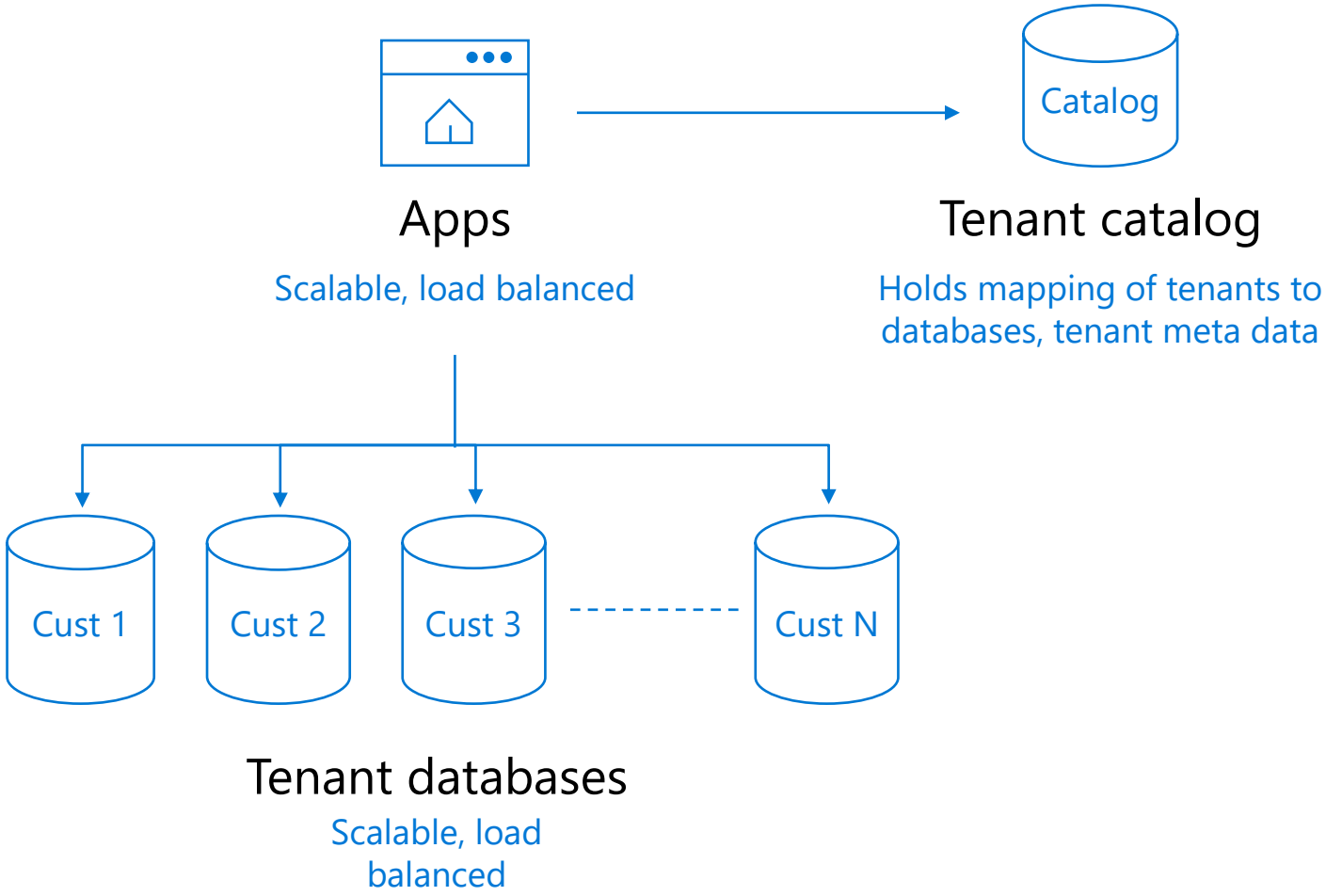
Multi-tenant data models



Multi-tenant data models – hybrid sharded



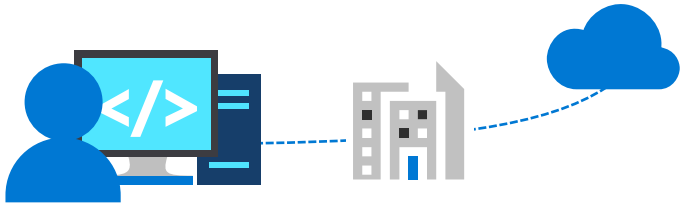
Canonical database-per-tenant SaaS app



Common app scenarios

Enterprise apps serving customers

Create new or re-architect apps to meet customer needs to augment their products and services



Cloud-born SaaS

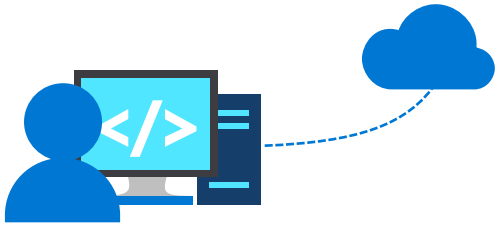
Disrupting traditional apps and business models

Common app types

Mobile/social
e-Commerce
Accounting
Payroll
Global marketing sites
Catalog
Gaming/voting
...and many others

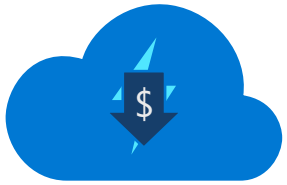
Enterprise apps serving employees

New or re-architected apps for internal use, reducing CAPEX and OPEX



ISVs going SaaS

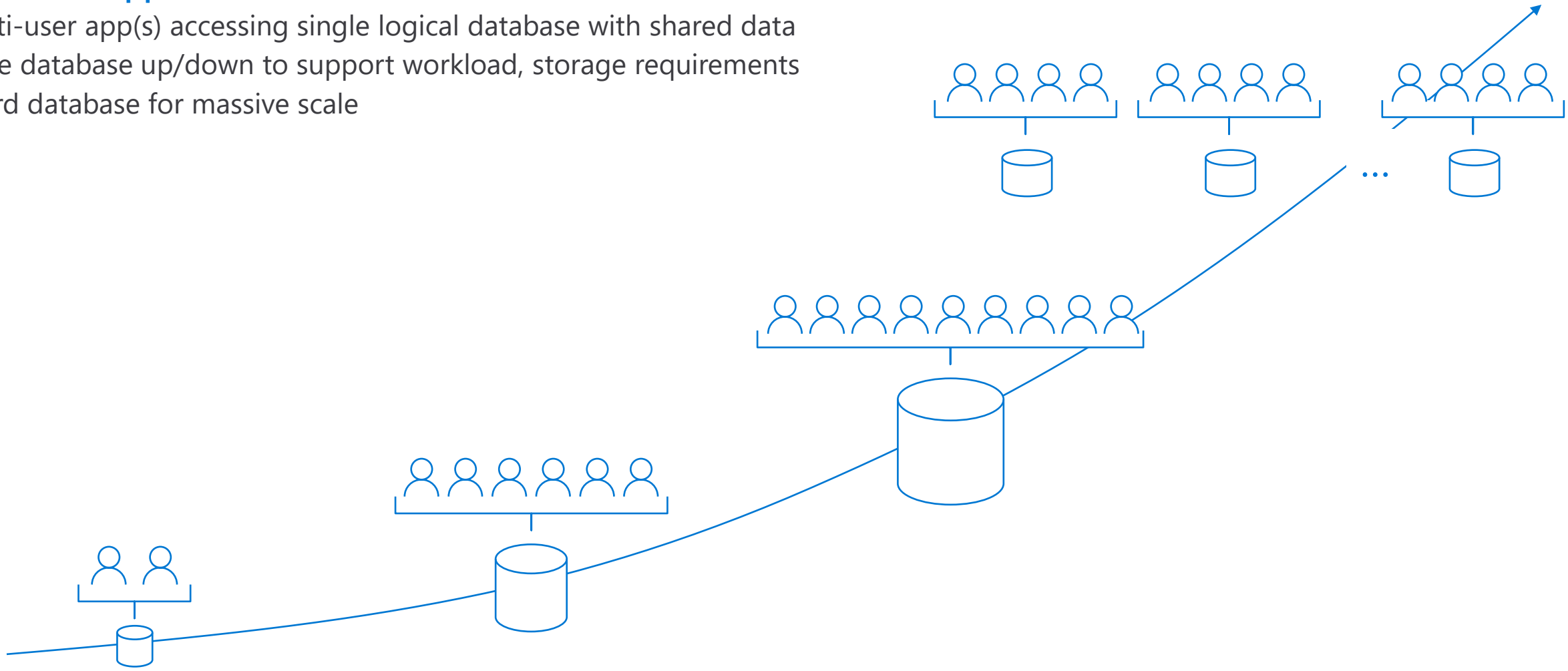
Re-architect applications to transition from selling software licenses to subscription-based SaaS



Enterprise cloud app workload patterns

Multi-user app, shared data

Multi-user app(s) accessing single logical database with shared data
Scale database up/down to support workload, storage requirements
Shard database for massive scale



SaaS app 1 workload patterns

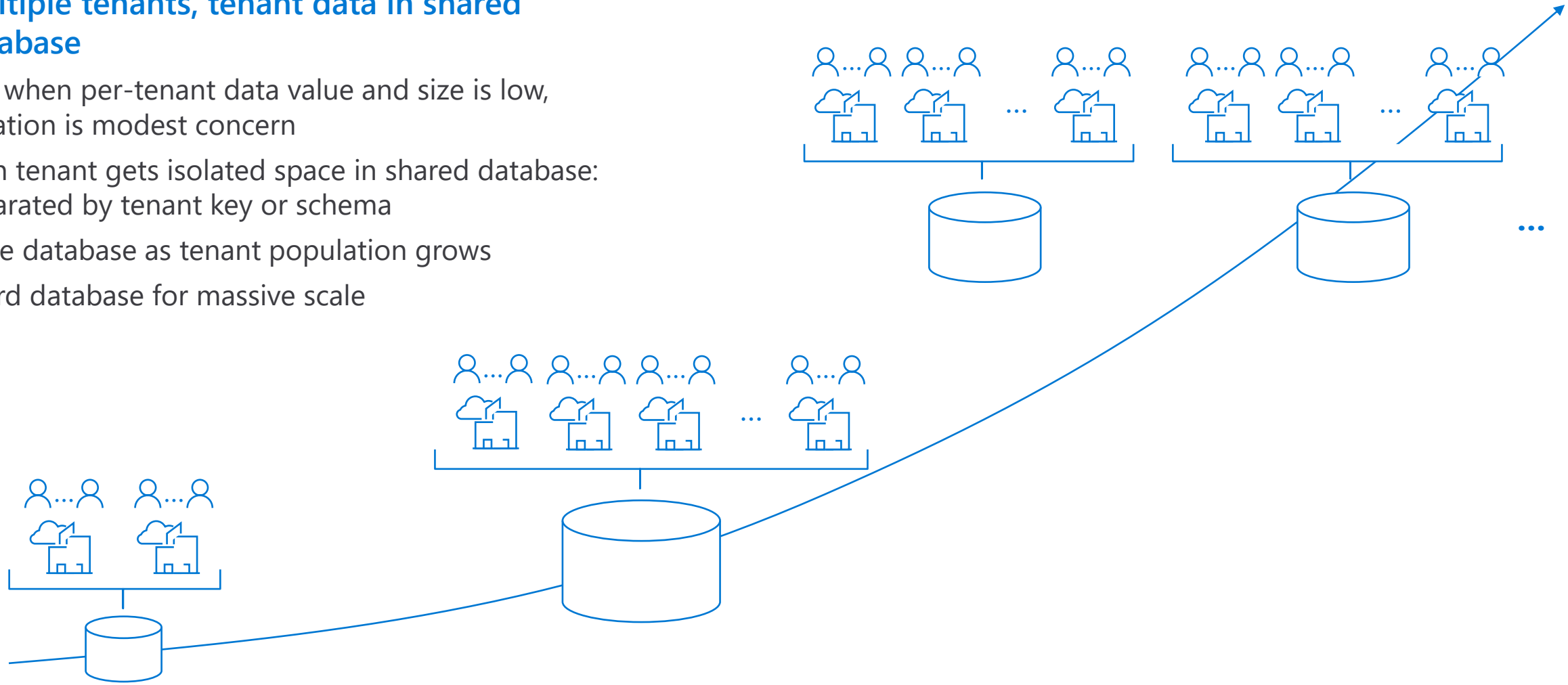
Multiple tenants, tenant data in shared database

Use when per-tenant data value and size is low, isolation is modest concern

Each tenant gets isolated space in shared database: separated by tenant key or schema

Scale database as tenant population grows

Shard database for massive scale



Outcomes from SaaS app 1 workload patterns

Requires multi-tenant-aware app logic and schema (tenant keys, RLS on tables, etc.)

Tenant data isolation is app responsibility

Additional management complexities, including:

Difficult to restore isolated customer
Difficult to monitor, audit individual customers
Shard management (e.g. split, merge)
Load balancing individual tenants
Handling churn
Custom schema difficult

SaaS app 2 workload patterns

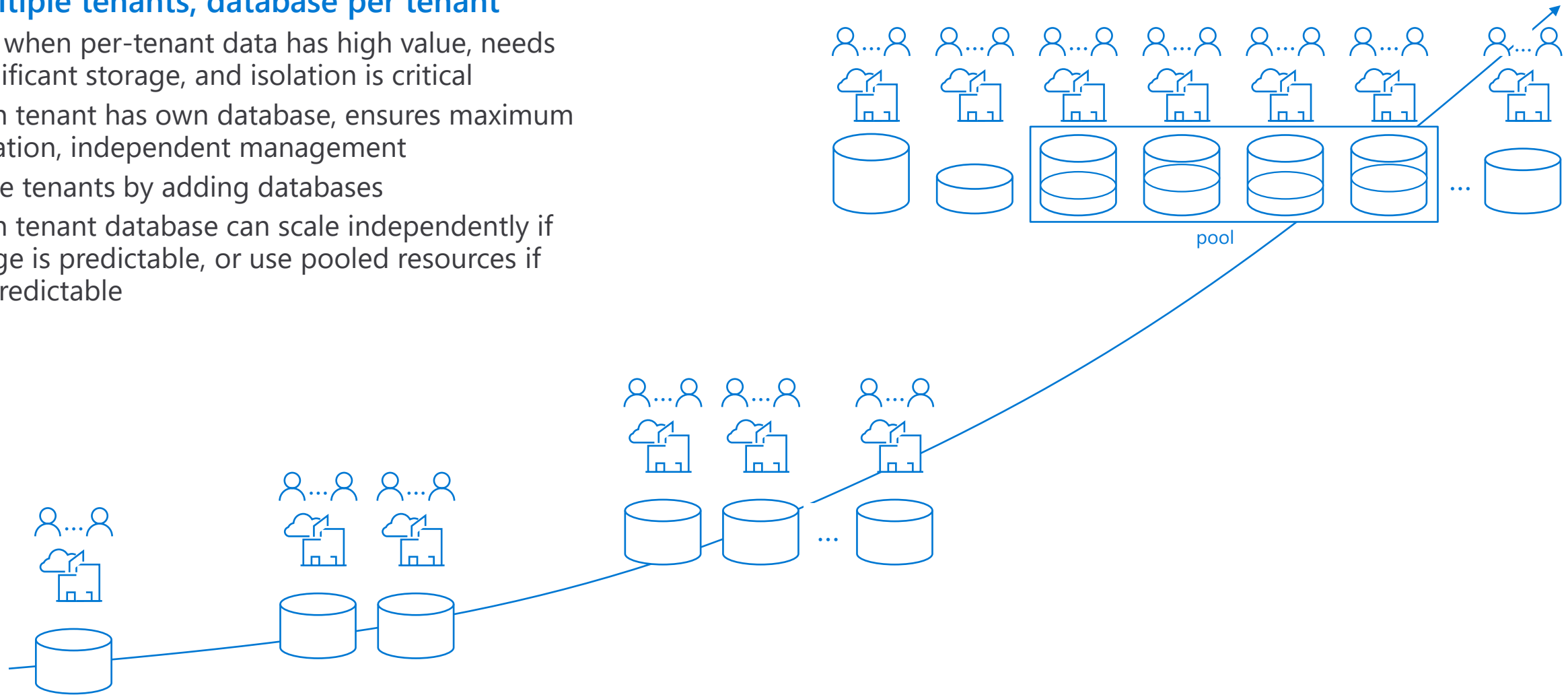
Multiple tenants, database per tenant

Use when per-tenant data has high value, needs significant storage, and isolation is critical

Each tenant has own database, ensures maximum isolation, independent management

Scale tenants by adding databases

Each tenant database can scale independently if usage is predictable, or use pooled resources if unpredictable



Outcomes from SaaS app 2 workload patterns

Simpler application logic

Straightforward cloud migration from existing app

Many databases, but management is often easier with:

Easy per-customer and pooled resource governance
Ability to restore single customer
Monitor, audit per-customer
Custom schema

Management at scale via scripts/jobs

Sharding and tenancy models

Single tenant per database

Each tenant's data is stored in a different database for better tenant isolation

Multiple tenants per database

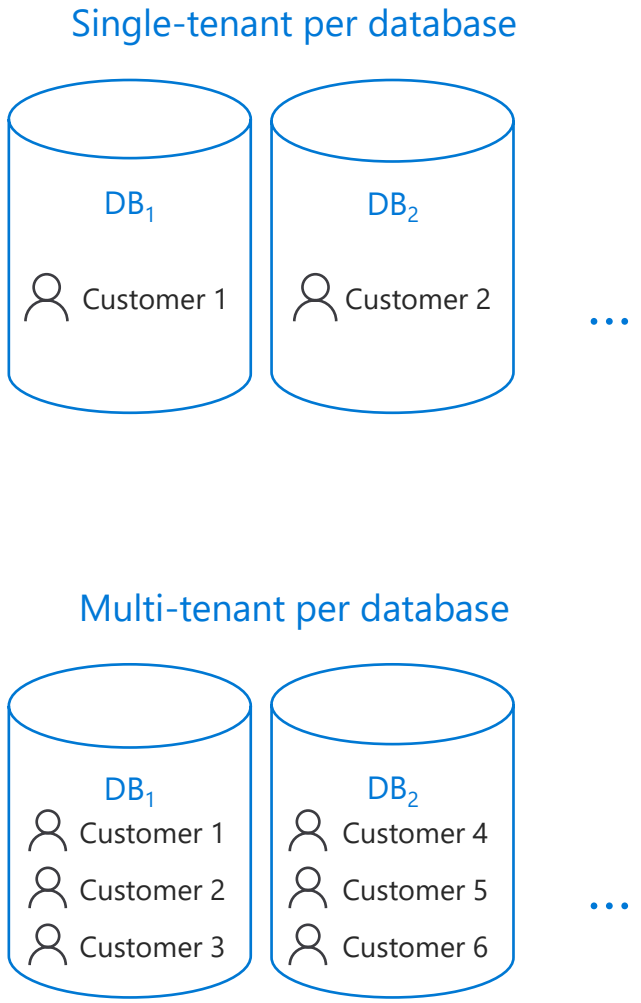
For less isolation of tenants as compared to single tenant model

Hybrid model

Some tenants share databases, others get their own database

Temporal model

Sharding based on date/time



High scale OLTP

Large, (mostly) uniform active data set

Distribute partition-able tables across shards as necessary

Some shards can become way more active than others

Pin hot users to dedicated databases

Move users to balance workload between databases

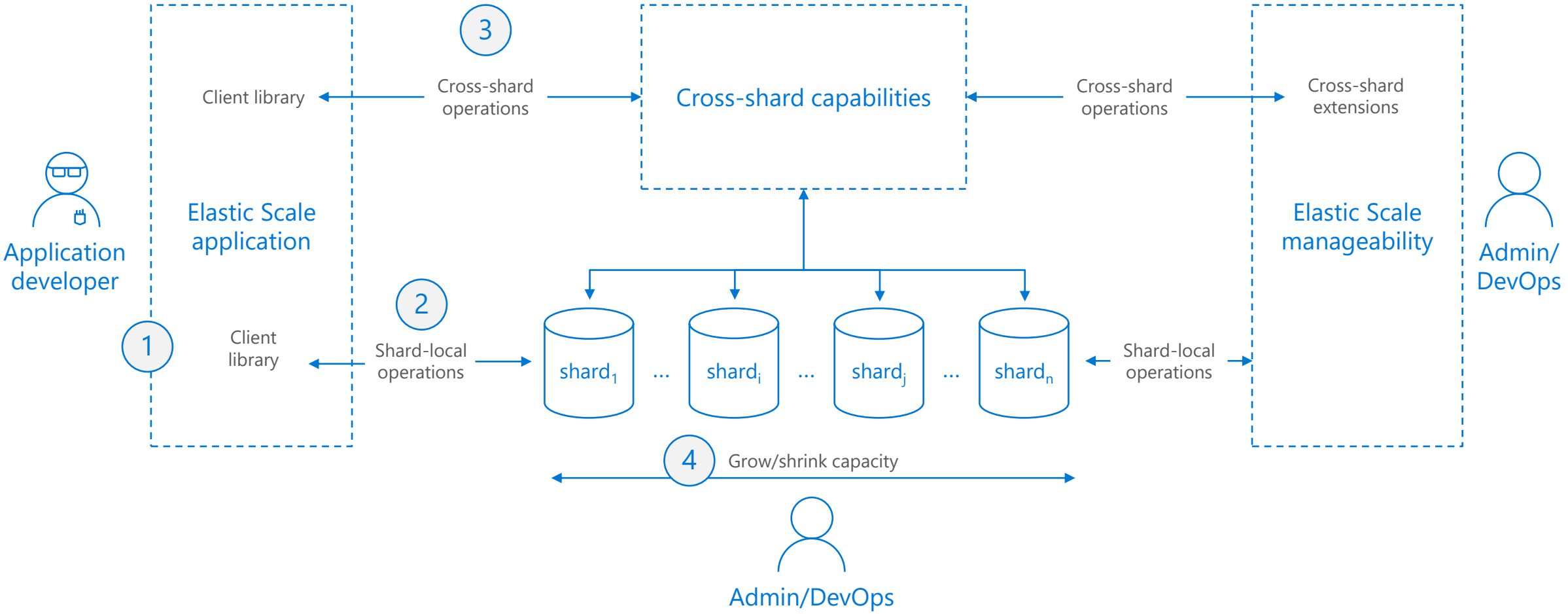
Scale to proper service tier to deal with increased workload

Split-merge actions are required to fully exploit elasticity

Mostly data dependent routing with key lookup queries

Few fan-out queries may be required (for example, leaderboards and inventory management)

Shard Elasticity



Microsoft Azure

Thank you.

Customer evidence

Learning Objectives

Common development scenarios

Customer stories

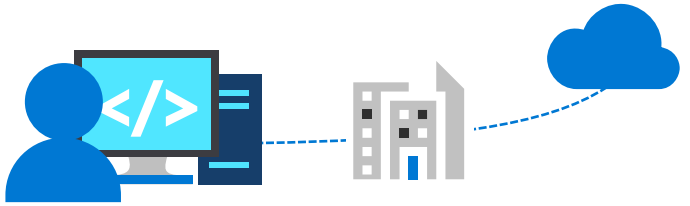
Common development scenarios

Cloud-born SaaS

Disrupt traditional apps and business models

Enterprise apps serving customers

Create new or re-architect apps to meet customer needs to augment their products and services

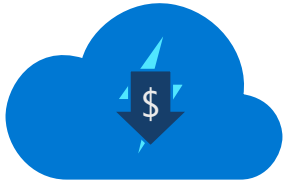


Common app types

Mobile/social app
e-Commerce app
Multitenant app
Global marketing sites
Catalog
Gaming/voting
...and many others

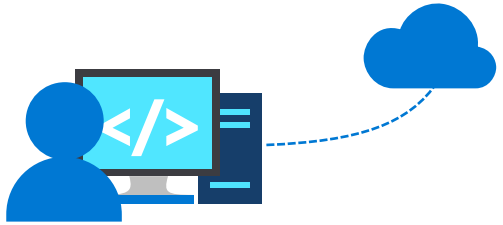
ISVs going SaaS

Re-architect for cloud to transition from selling software licenses to subscription-based SaaS provider



Enterprise apps serving employees

Create or re-architect apps to cloud for internal use, built by internal development staff



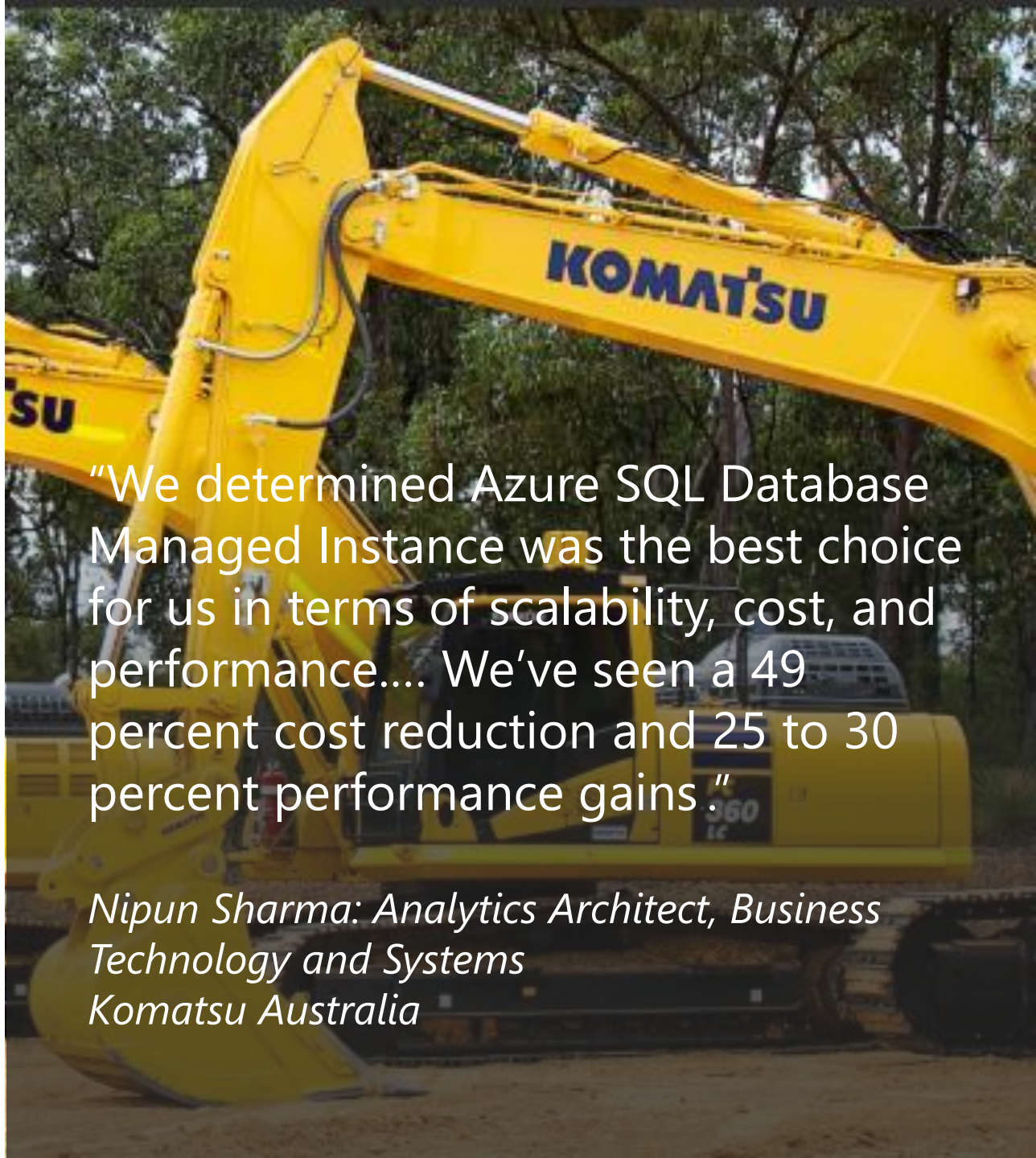
Komatsu achieves breakthrough performance gains and savings with Azure SQL Database Managed Instance

Challenge

Had multiple mainframe applications for different types of data, and needed to consolidate them into a single system so that different parts of the organization can easily get a holistic view of data from different sources

Impact

Moved the entire database and the Discovery Hub analytics implementation to Azure SQL Database Managed Instance in less than two weeks, and since then we've seen a 49 percent cost reduction and 25 to 30 percent performance gains."



"We determined Azure SQL Database Managed Instance was the best choice for us in terms of scalability, cost, and performance.... We've seen a 49 percent cost reduction and 25 to 30 percent performance gains."

Nipun Sharma: Analytics Architect, Business Technology and Systems
Komatsu Australia

World-class, cloud-enabled time and labor application.

Challenge

Flex Time by Paychex handles 13 million timecards and 3 million distinct punches on any given day. With an increasing volume of customers, however, managing that environment became very cumbersome. Guaranteeing uptime meant constant investments in new hardware, which was both time-consuming and capital-intensive.

Impact

Paychex engineers are freed from managing infrastructure, licensing, and database spin up and spin down. Built-in intelligent features like automatic tuning optimize database performance, leaving them to focus on what they do best: understand how clients are accessing the app and provide value in their continuous delivery pipeline.

PAYCHEX
Payroll | Benefits | HR | Insurance

Transitioning to an all-cloud IT environment

Phased, policy-based migration to Azure

Challenge

Organization had over 30 terabytes of unstructured data on 15 on-premises file servers, some of it more than 12 years old

Wanted to eliminate content clutter and move to a cloud-only IT strategy

Impact

Decommissioned more than 80 servers and eliminated the need to invest in new equipment and tools

Saved over $20,000 a month in maintenance costs

Cut archive costs by two-thirds

Reduced 30 terabytes of data on file servers to less than 8 terabytes on Azure

Processing invoices 24 times faster with the cloud

Moved eBillingHub to the Azure cloud

Challenge

Electronic invoicing solution was unable to scale as needed to accommodate increased invoice submissions during billing cycles

Impact

Processed 2.5 million invoices in the first three months after launching

Reduced time to process 28,000 invoices from 2 data to 2 hours

Increased transaction speed to 74 validations per second

THOMSON REUTERS®

Managing and growing assets worth €350 million

Hosted management service on Microsoft Azure

Challenge

Financial startup needed to build an enterprise platform that could support their algorithms, meet compliance, and safeguard data

Wanted easy scalability and reliable business continuity

Impact

Ability to update and enhance services faster and easier, at less cost

Scalable compute and storage allows Metori to manage €350 million in assets, trade €30 billion a year, and execute transactions worth €150 million a day

METORI
Capital Management

"Without Azure, it might have cost us 10 times more to build an environment that could cope with the future we expect."

Nicolas Gaussel
CEO, Metori Capital Management

Real-time product recommendations for 15.4m customers

Relational customer data stored in Azure SQL DB

Challenge

Global online fashion retailer wanted to provide a more personalized shopping experience and speed order updates

Needed to make real-time product recommendations for their 85,000 items with 5,000 new added each week

Impact

Real-time product recommendations and instant order updates for 15.4 million customers

Served 167 million customer requests in 24 hours, handling 3,500 requests and 33 orders a second with an average response time of 48 milliseconds

asos
discover fashion online

Enhancing the coverage of Virginia elections

Real-time election updates using Power BI

Challenge

Media agency wanted an effective way to visualize election results

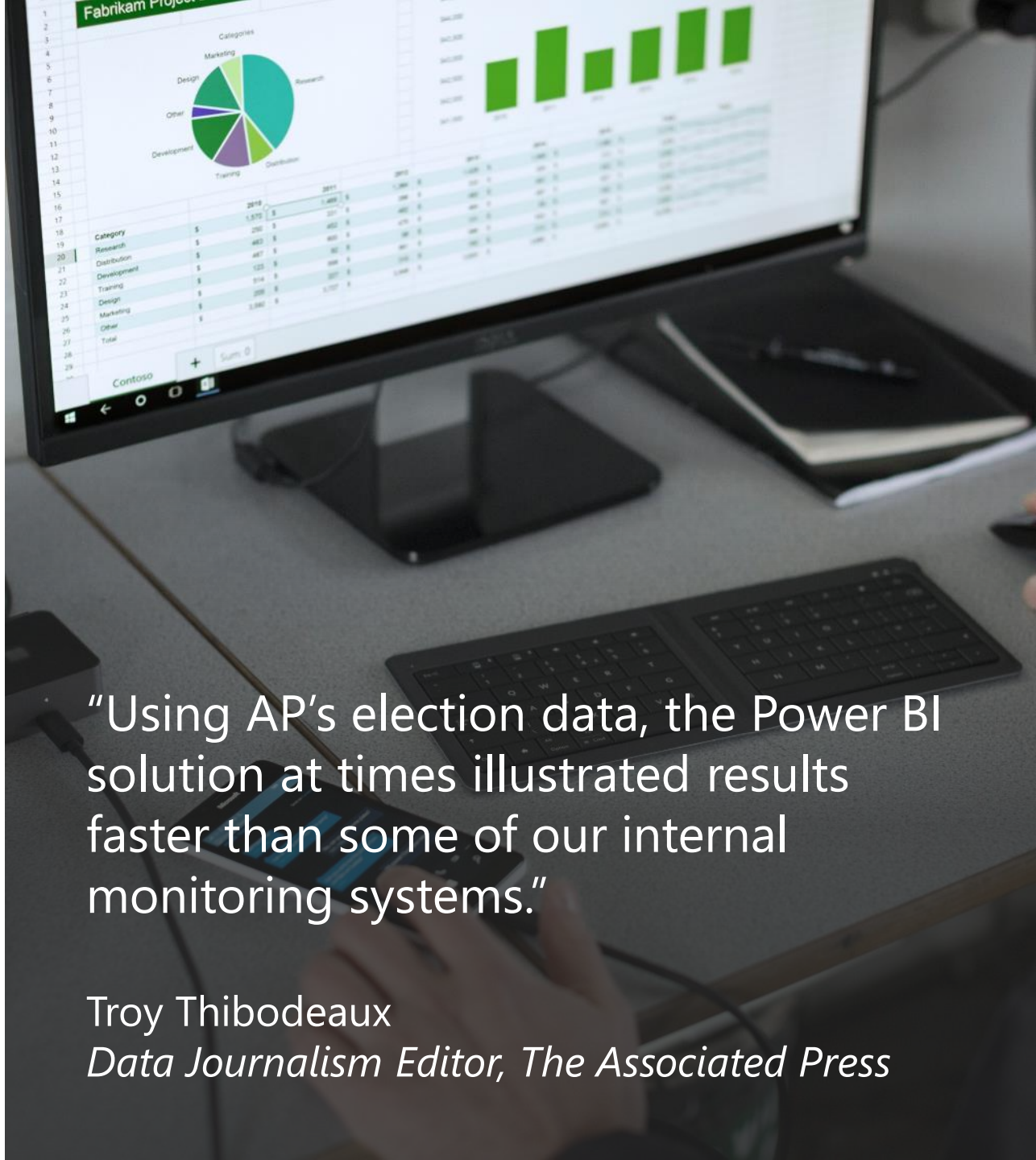
Needed to maintain the up-to-the minute coverage that they're known for during key races

Impact

Created compelling and interactive live data visualizations to share news more quickly

Ability for readers to interact with data themselves, regardless of screen size

Live election night results visualization illustrated results faster than AP's own internal monitoring tools



"Using AP's election data, the Power BI solution at times illustrated results faster than some of our internal monitoring systems."

Troy Thibodeaux
Data Journalism Editor, The Associated Press

Next-generation enterprise content management

Elastic database pools adjust to shifting customer needs

Challenge

Enterprise content management vendor had too many different content systems and storage destinations

IT administration was acting as an obstacle to development

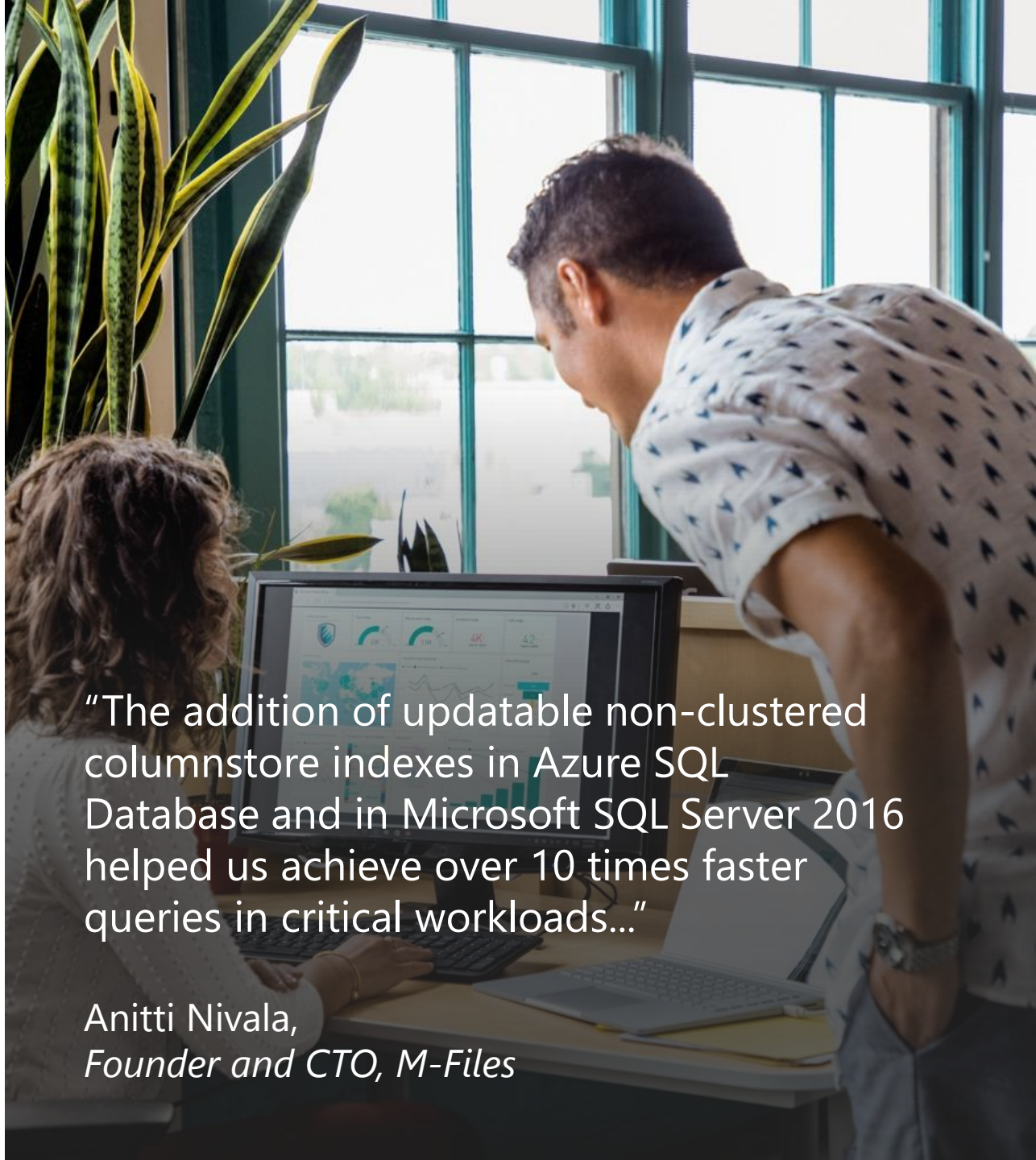
Impact

Experienced 6x faster growth than the pace of the ECM market

10x faster queries in critical workloads

1,000 databases in 12 data centers cost 3-10x less by using elastic database pools instead of standalone database instances

M-Files®



Serving the latest tech news to 2 million readers

Migrated WordPress site to Microsoft Azure platform

Challenge

As popularity and site traffic increased, performance issues emerged that were difficult to diagnose

Unpredictable 5-10x increases in traffic required the ability to scale on the fly with no downtime for the website

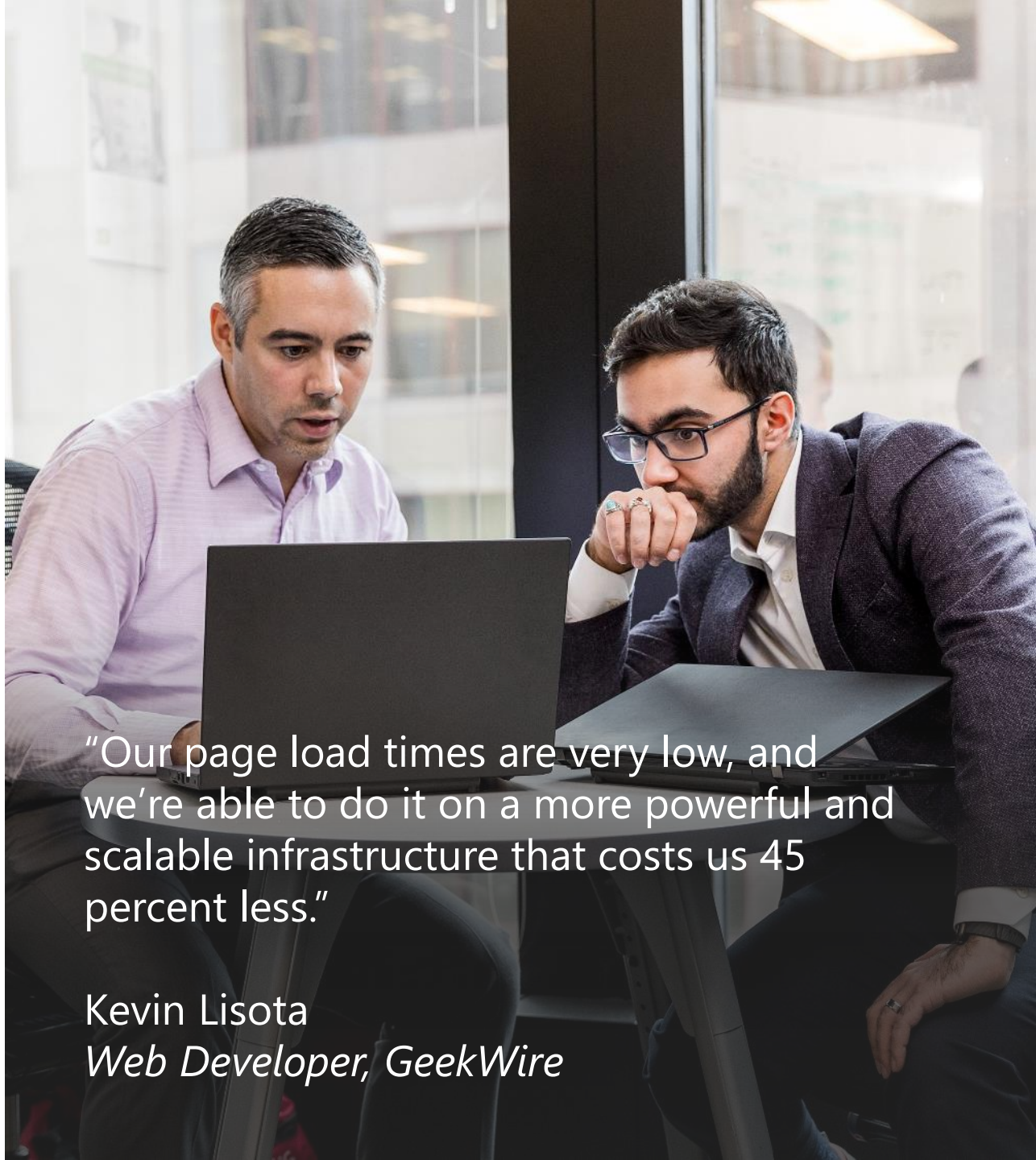
Impact

Cut costs by 45% while still scaling on-demand

Ability to support more traffic at a lower cost

Extremely low page load times

GeekWire



"Our page load times are very low, and we're able to do it on a more powerful and scalable infrastructure that costs us 45 percent less."

Kevin Lisota
Web Developer, GeekWire

Doubling sales and handling up to 33 orders per second

Migrated SQL-based commerce platform to Azure

Challenge

Online retailer needed an app that could scale to meet seasonal demand

Existing on-premises infrastructure risked a single point of failure

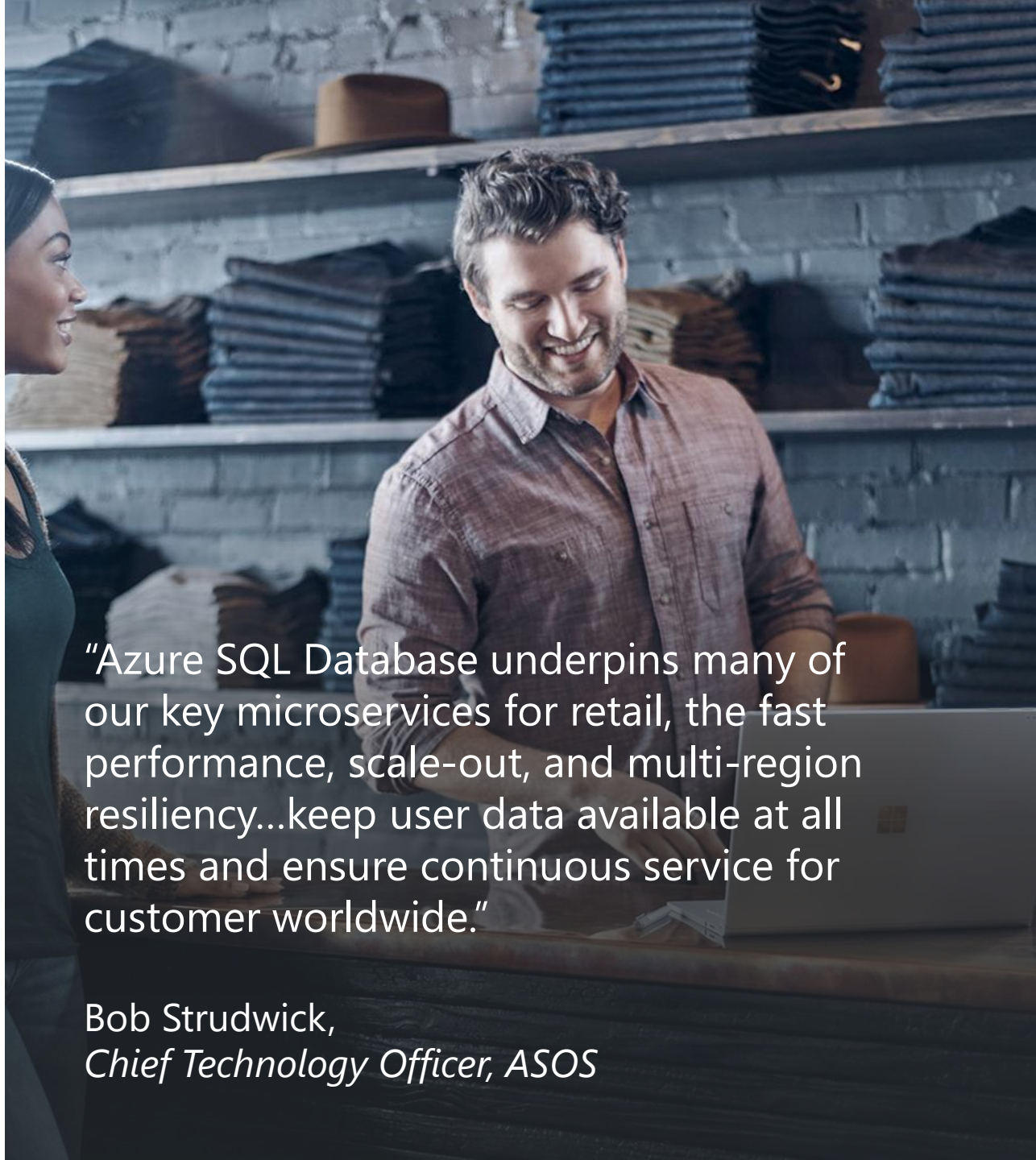
Impact

30 globally distributed datacenters ensure high levels of performance and availability for customers in any location

Ability to easily tailor front-end services without modifying back-end infrastructure

30% revenue growth year after year with more responsive apps that deliver personalized customer experiences

asos
discover fashion online



Doubling key database's workload for less

Memory-optimized tables in elastic pools

Challenge

ISV needed massive scale to monitor oil and gas data in real-time

Current rate of innovation far exceeds a traditional infrastructure model

Database that stores incoming data from customers' devices was nearing its data-ingestion limits

Impact

70% reduction in DTU consumption

Memory-optimized tables manage dramatic and unpredictable spikes in demand

quorum

Combing and analyzing 150k articles daily

Real-time media monitoring with Azure SQL Database

Challenge

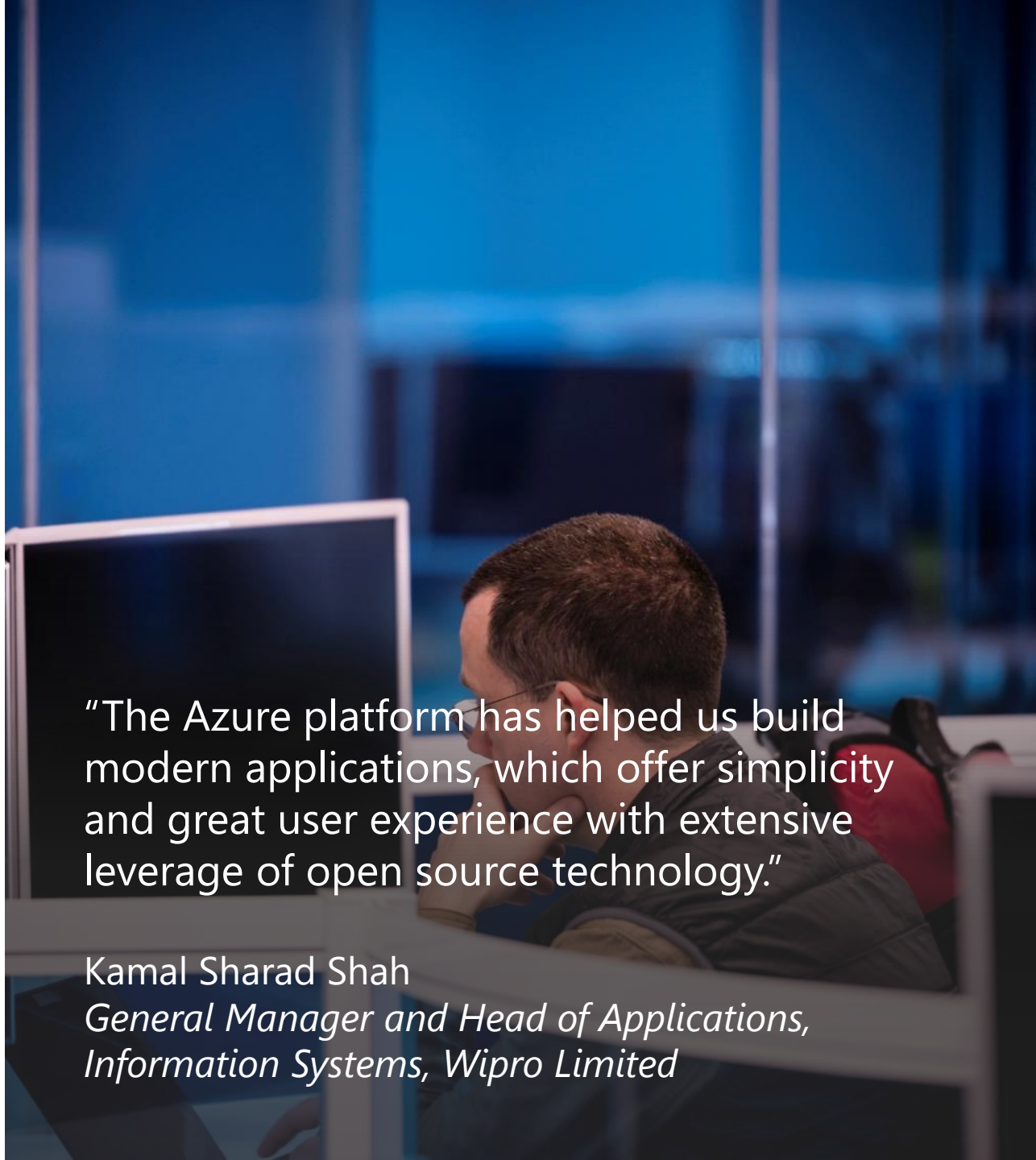
IT company was seeking a cost-effective, resilient, and enterprise friendly IT infrastructure solution

Impact

Developed and deployed an entire application in just 6 weeks

Decreased lines of code by 10x with extensive use of micro services from Azure PaaS

Estimated 60% cost savings on developing and deploying a new app



Creating greener solutions with open source on Azure

Open source software and big data

Challenge

HVAC equipment provider wanted to create an easier, more automated way to aggregate data and provide detailed intelligence on systems running worldwide

Needed seamless implementation of on-premises Linux-based clusters

Impact

Hyper-scalability and support for open source software enabled global rollout

Optimal flexibility, fast time-to-market, and improved energy efficiency

Connected over 5,000 rooftops and 40,000 subsystems to Azure

Johnson Controls

"The adaptability, security, scalability, and reliability of Microsoft's platform along with the support for open source appeals to us. We like working with Microsoft—it's a partnership."

Sudhi Sinha,
VP, Product Development, Johnson Controls

100 Trillion transactions per day

Scaling an e-commerce platform with a data-powered customer experience

Challenge

Online marketplace was unable to build and manage the datacenters and development infrastructure to meet their growth strategy

Unpredictable customer traffic required rapid and flexible scaling

Goal to become one of the leading e-commerce destinations in just 18-36 months

Impact

50% faster time to market

Ability to scale across regions, processing 100 trillion transactions per day

20TB of data analyzed per day to monitor price history and market dynamics

jet

Azure SQL Database wrap-up

Microsoft Azure

Azure SQL Database — Everything built-in

The intelligent relational cloud database service

Intelligent performance

Realize automatic performance improvements from continuous assessment and innovation

Scales on the fly

Change service tiers, performance levels, and storage dynamically without downtime

Business continuity

Easily manage and monitor business critical functions for reliable operations

Works in your environment

Develop your app and connect to SQL Database with the tools and platforms you prefer

Advanced threat protection

Build security-enhanced apps with built-in protection and industry-leading compliance

Learn more

Pricing: https://azure.microsoft.com/en-us/pricing/details/sql-database/managed/

Services: https://azure.microsoft.com/en-us/services/sql-database/

Documentation: https://docs.microsoft.com/en-us/azure/sql-database/

Stack overflow: http://stackoverflow.com/questions/tagged/sql-azure

Microsoft Azure