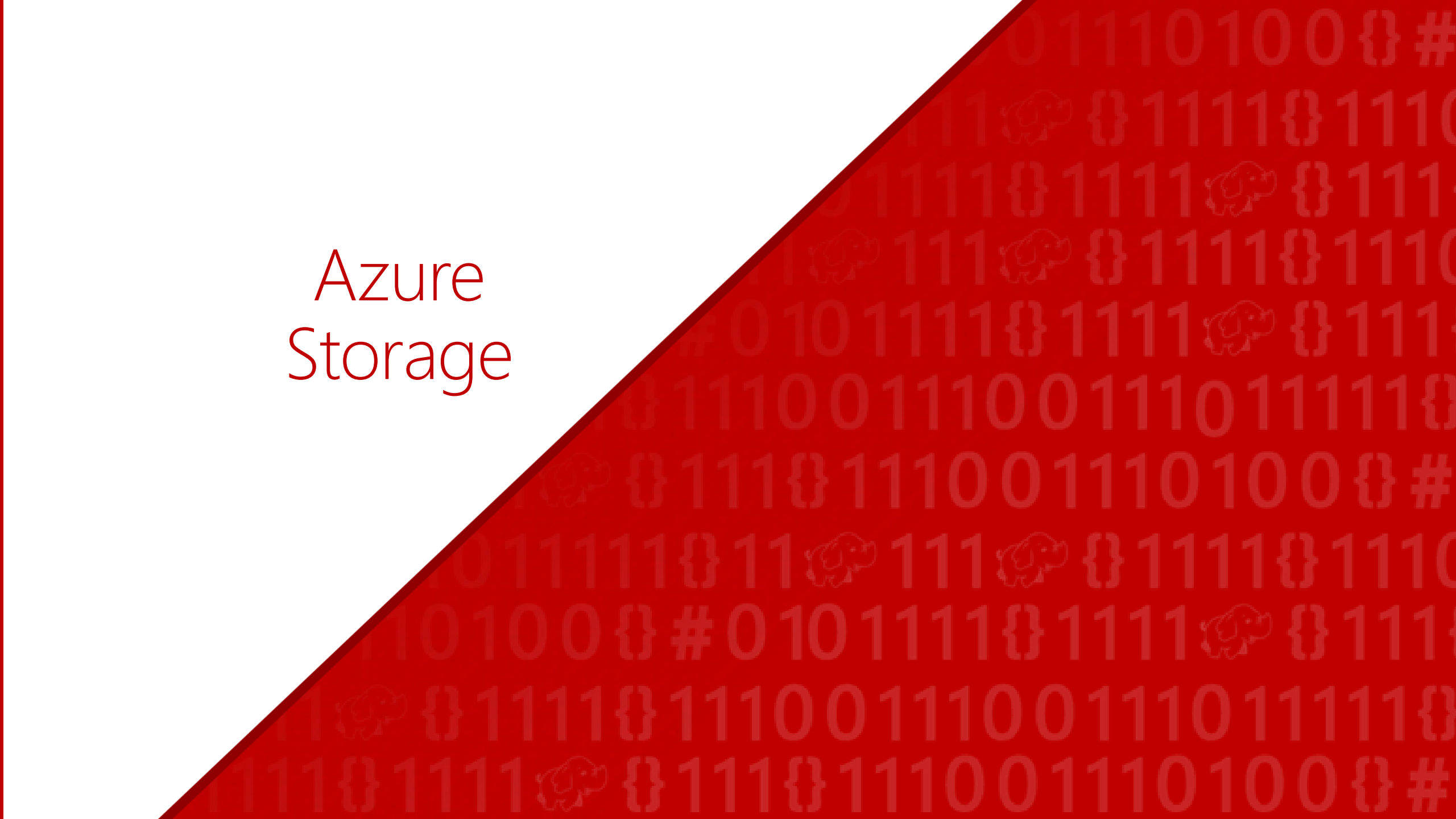
Azure Machine Learning Storage

Marcin Szeliga

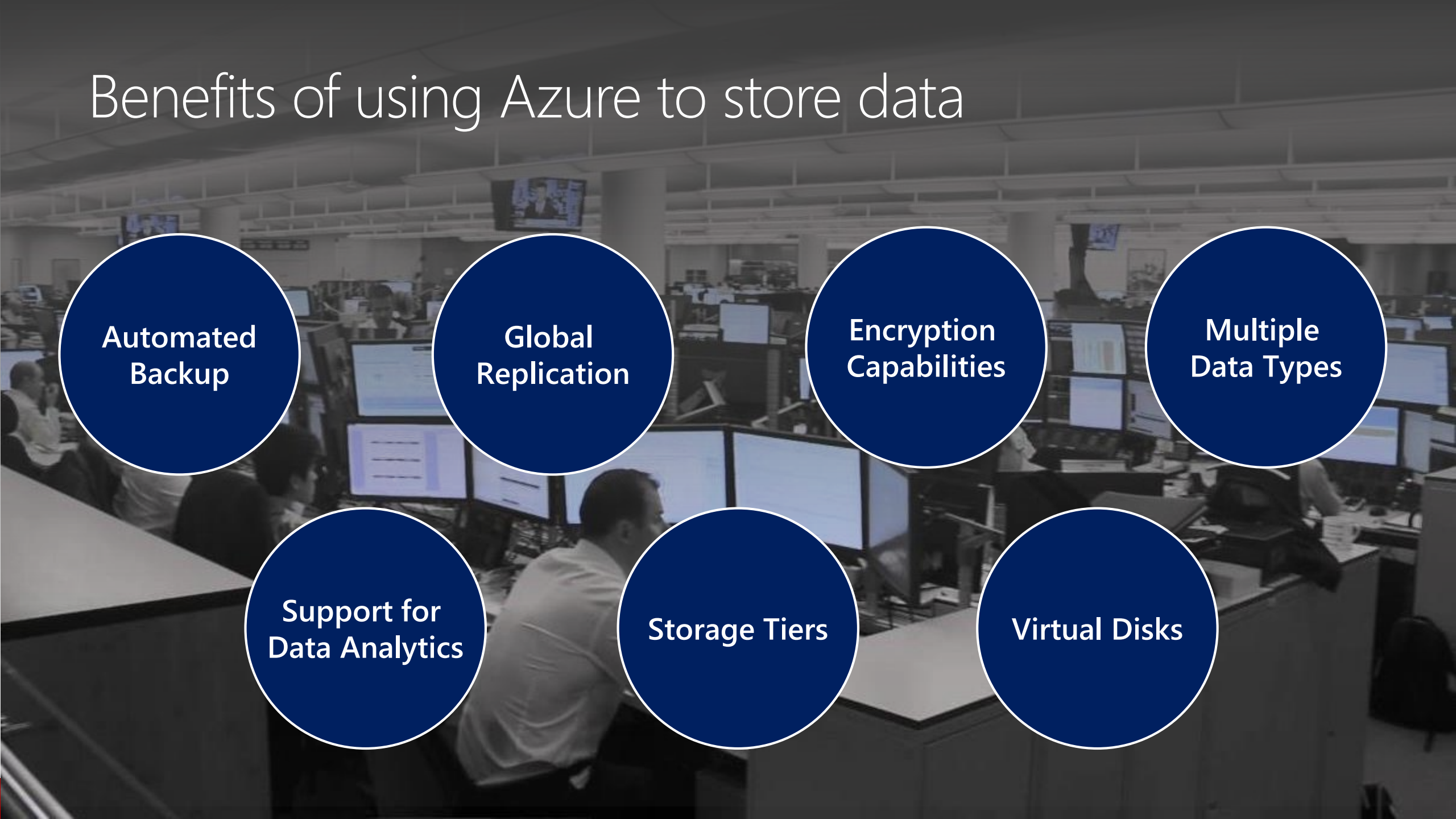
Agenda

- Azure Storage
- Data Lake Storage
- Cosmos DB

Azure Storage



Benefits of using Azure to store data



Storage accounts

What is a Storage Account

It is a container that groups a set of Azure Storage services. Only data services can be included in a storage account such as *Azure Blobs, Azure Files, Azure Queues, and Azure Tables*.

How many do you need?

The number of storage accounts you need is typically determined by your data diversity, cost sensitivity, and tolerance for management overhead.

The number of storage accounts you need is based on:

Data Diversity

Organizations often generate data that differs in where it is consumed and how sensitive it is.

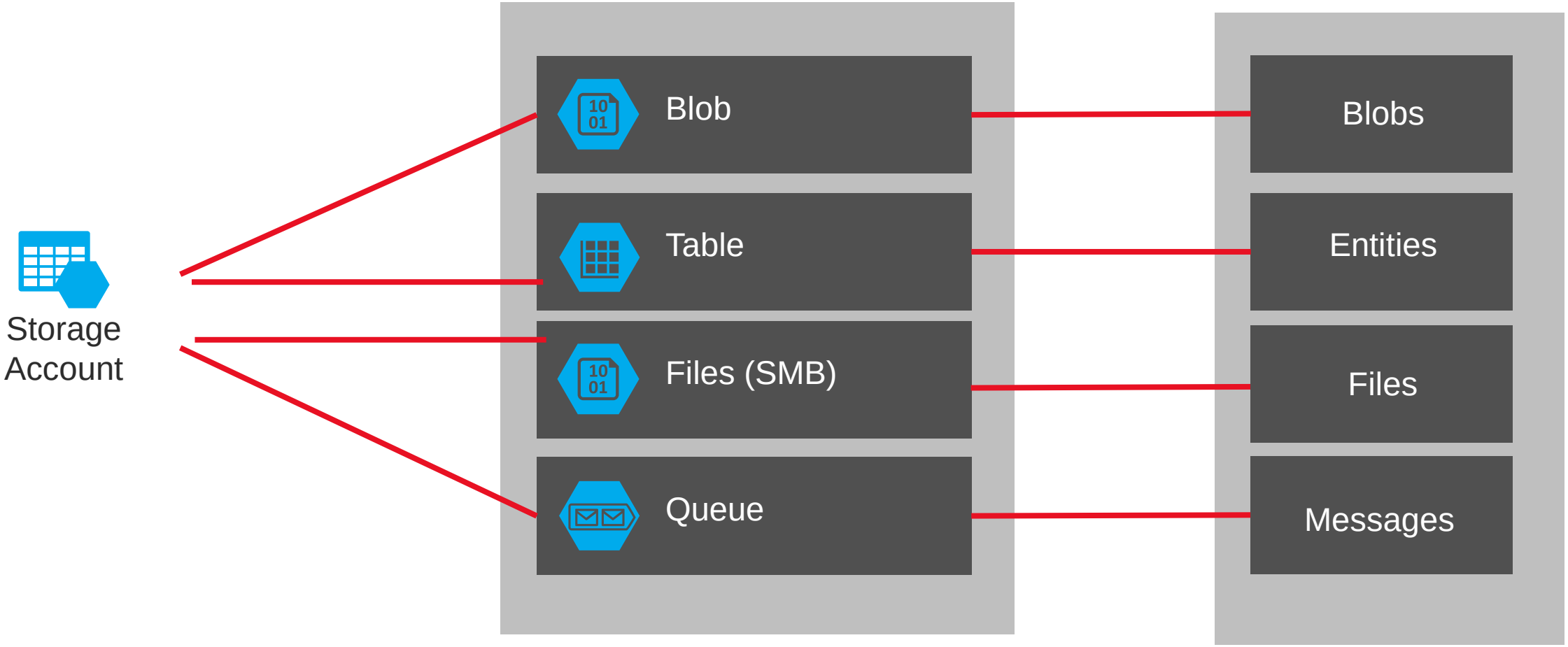
Cost Sensitivity

The settings you choose for the account do influence the cost of services, and the number of accounts you create

Management Overhead

Each storage account requires some time and attention from an administrator to create and maintain.

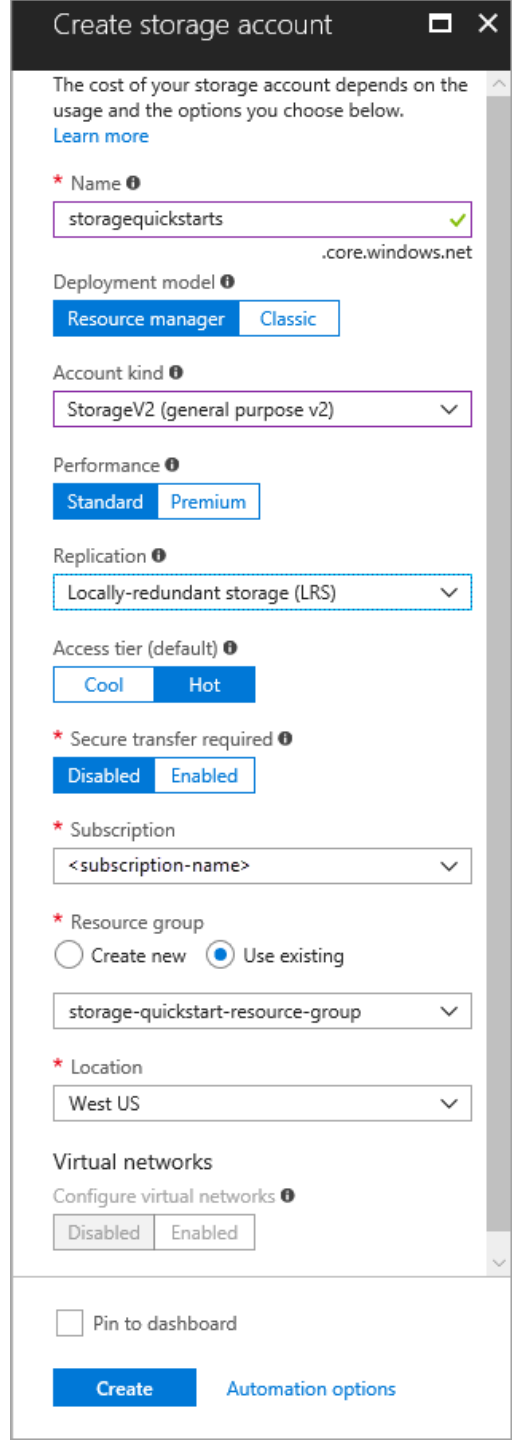
Azure Storage Services




```
http://<storage acct>.blob.core.windows.net/<container>/<blob>
http://<storage acct>.table.core.windows.net/<table>
http://<storage acct>.queue.core.windows.net/<queue>
http://<storage acct>.file.core.windows.net/<file>
```

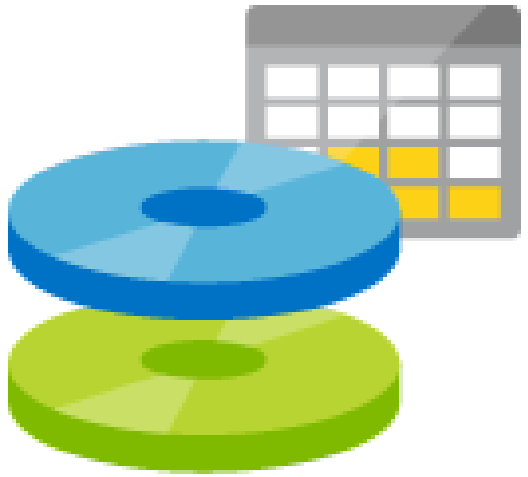
Three types of Storage Accounts

- General purpose v1
  - Tables, disks, queues, files and blobs
  - Standard and premium tiers
- Blob storage
  - Specialized storage for binary large objects
  - Two access tiers: hot and cool storage
  - Recommended for applications that need only this kind of storage
- General purpose v2
  - Support all APIs and features supported in GPv1 and Blob storage accounts
  - Up to 500 TB of data in a single storage account



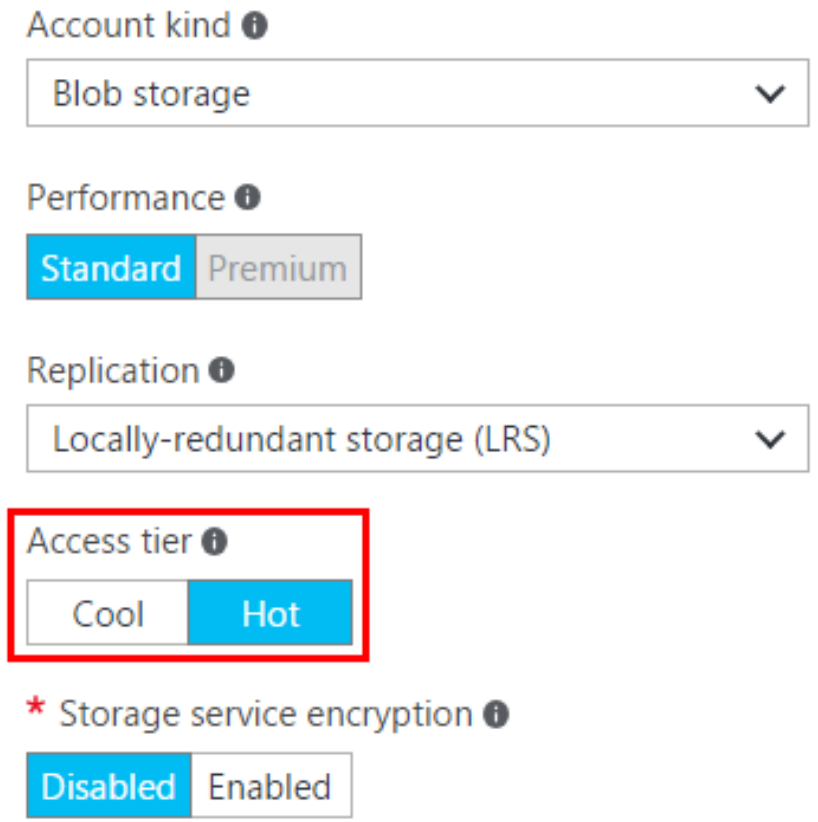
Standard and Premium Accounts

- Standard
  - Based on magnetic drives
  - Max throughput per disk – 500 IOPS, 60 MB/s
  - Up to 20 000 IOPS
- Premium
  - Based on SSD
  - Max throughput per disk – 5000 IOPS, 200 MB/s
  - Up to 50 Gbps (all disks)
- Apply to General storage accounts



Access Tiers in Blob Storage Accounts

- Hot storage
  - Frequently accessed data
  - Higher storage costs
  - Lower access and transaction costs
  - Higher SLA (99,9% / 99,99% availability)
- Cool storage
  - Infrequently accessed data such as backups, security camera foota
  - Lower storage costs
  - Higher access and transaction costs
  - Slightly lower SLA (99% / 99,9%)



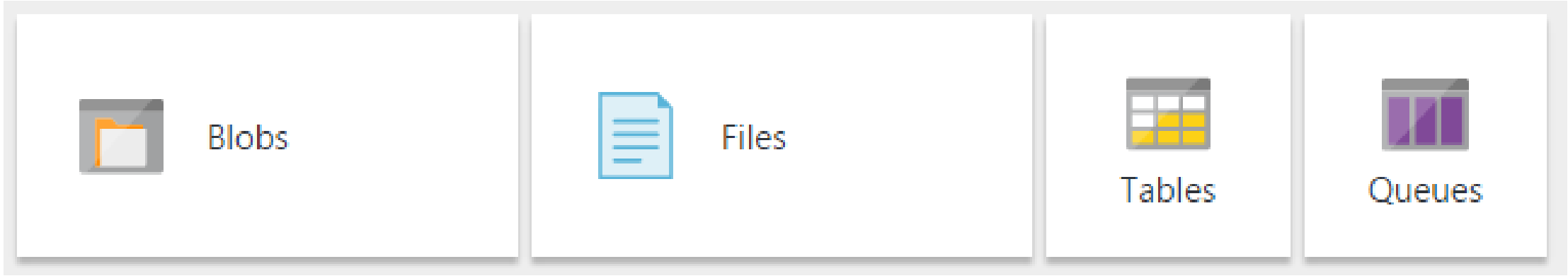
Azure Storage

Blob Storage
Storage of files, VHDs, mp4s, pngs, .txt, etc…

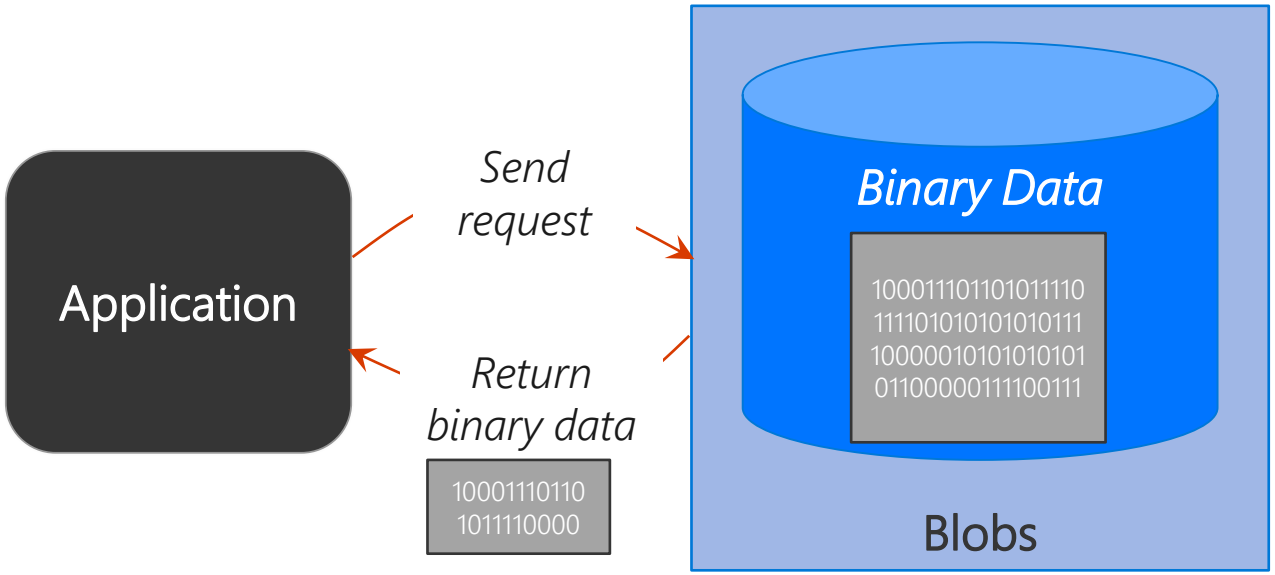
Files
SMB File Sharing

Tables
Highly scalable service for non-relational structural data

Queues
Low latency message processing

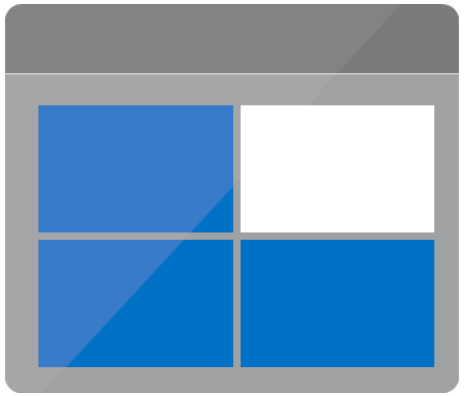


Azure Blobs
A fundamental data service



Azure Blobs
A fundamental data service

- Storage for arbitrary unstructured data
  - Backups, documents
  - Movies, music, images
  - Logs, VM images, big data
- Massive scalability
  - Automatically scales up/down as required
- Tiered storage options
- Object mutability
  - Edit objects in place – performance & bandwidth gains



Why Use Blobs?

Broadly useful

Can store pretty much anything

Also provides *cool storage* and *archive storage* tiers

Scalable

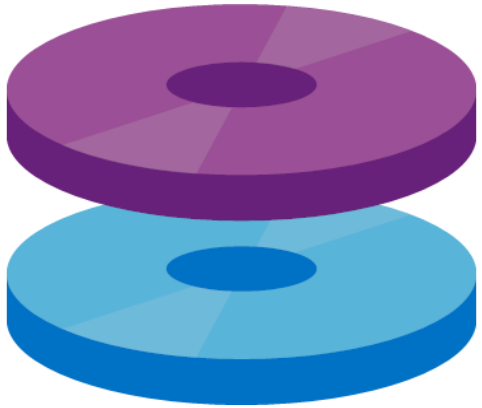
A single blob can hold up to 200 gigabytes

Low cost

Can cost just a few cents per gigabyte per month

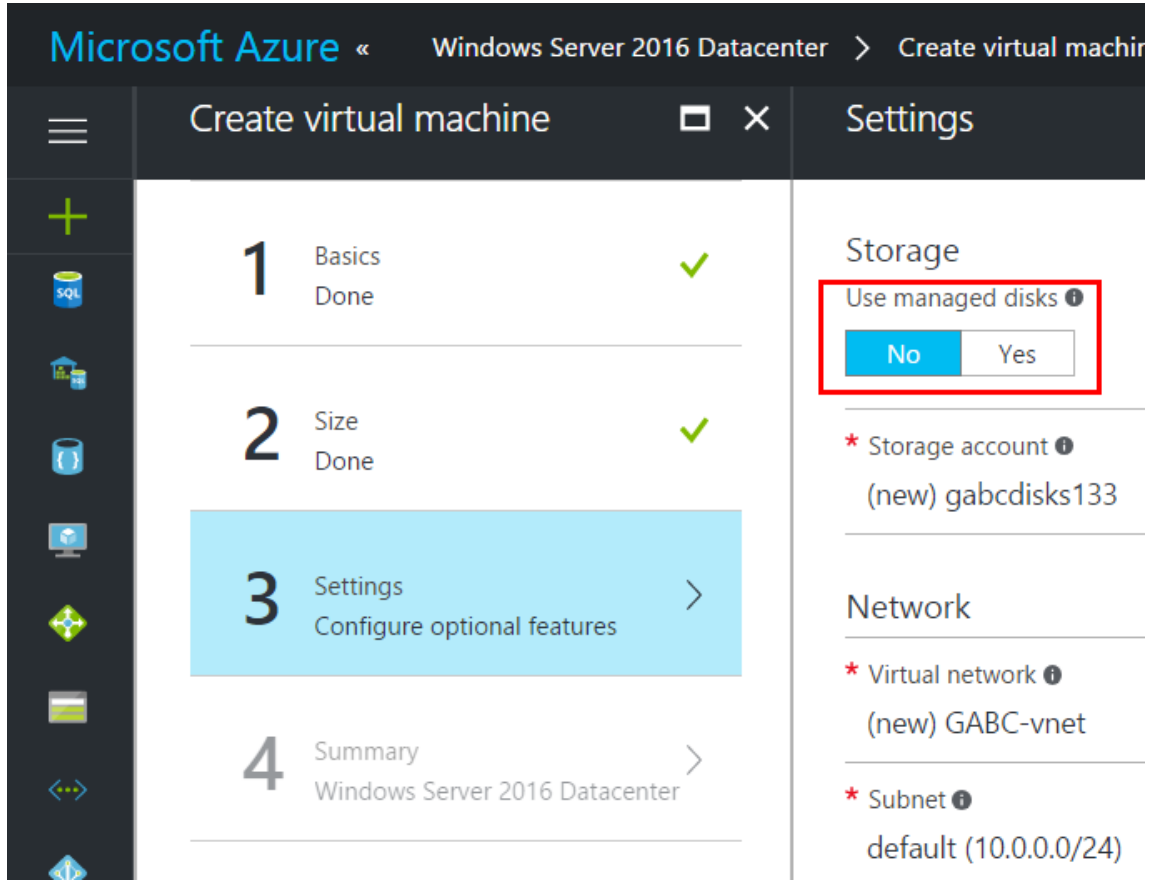
Disk Storage

- Storage for Virtual Machines
- Low latency, high throughput
- Standard – HDD, Premium – SSD
- 3 replicas of data
- Azure Disk Encryption
  - OS volume, data volumes
- Cross platform – Linux support
- SQL Server, Dynamics AX, Exchange Server, SharePoint certified with premium storage



Disk Storage

- Managed disks
  - 32 GB - 1 TB (S), 128 GB – 1 TB (P)
  - Handle storage automatically
  - Up to 10000 disks
  - Central management
  - Easy scaling (e.g. no 20000 IOPS storage account limit)
  - Point in time backups (snapshots)
  - Better reliability for Availability Sets
  - Granular RBAC
- Unmanaged disks
  - Legacy VM disk storage method
  - Manual Storage Account management
  - Not recommended



Azure Tables
A key/value store

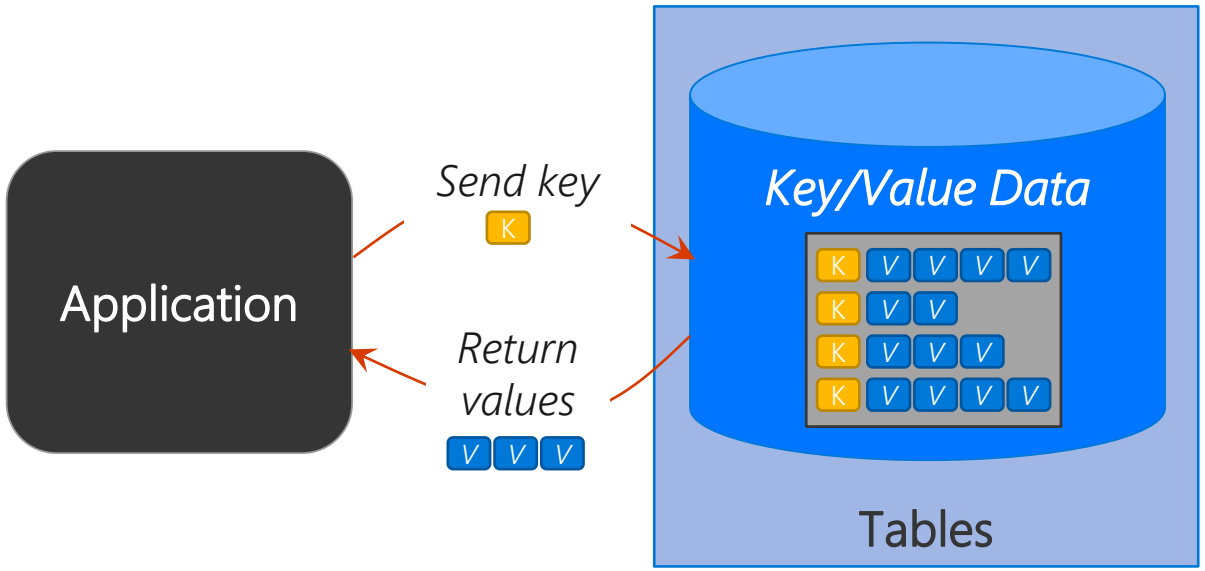


Table Storage

- A NoSQL schemaless store
- Massively scalable, highly available
- Storage in the form of tables
- Table is a collection of entities with properties and their values
  - Each entity (“row”) can have different properties
- Tables are partitioned
  - Partition key + row key -> primary key
- Table service doesn’t require any schema
  - Schema may be created and enforced on the client side

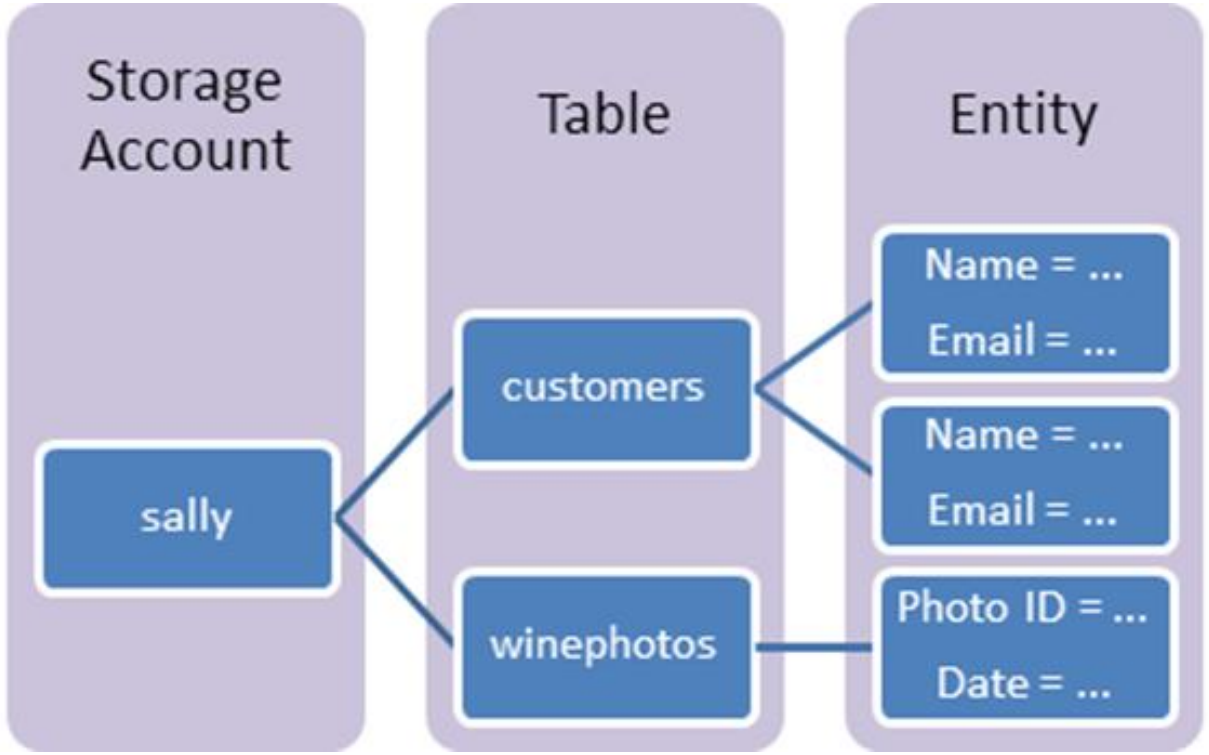


Table Storage

- Accessible via REST API
- Supports OData queries
  - $filter, $top, $select OData query options
  - JSON and AtomPub formats
- Supports Linq

```
https://myaccount.table.core.windows.net/Customers()?$top=10

https://myaccount.table.core.windows.net/Customers(PartitionKey='S',
RowKey='SmithJohn')

https://myaccount.table.core.windows.net/Customers()?$filter=LastName
%20eq%20'Smith'%20and%20FirstName%20eq%20'John'
```

Table Storage

Example queries

Node.js

```
var query = new azure.TableQuery()
            .select(['description', 'dueDate'])
            .top(5)
            .where('PartitionKey eq ?', 'hometasks');

tableService.queryEntities('MyTable', query, ...);
```

C#

```
var query = from entity in table.CreateQuery<Customer>()
            where entity.AmountDue <= 100.25
            select entity;
```

Why Use Tables?

Simple to use

Key/value stores provide a straightforward data model

Very scalable

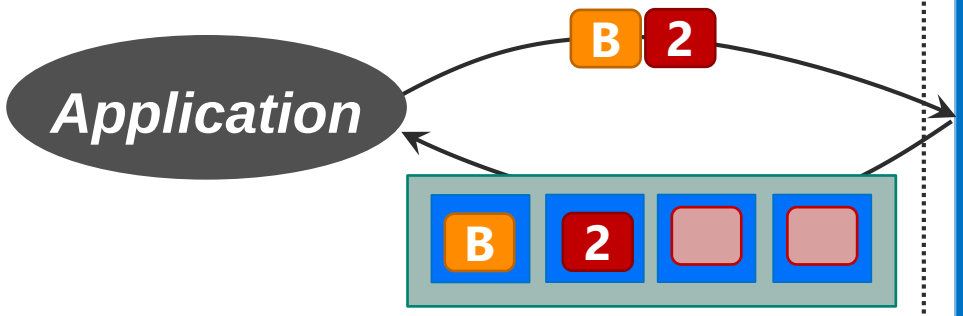
A single database can hold hundreds of terabytes

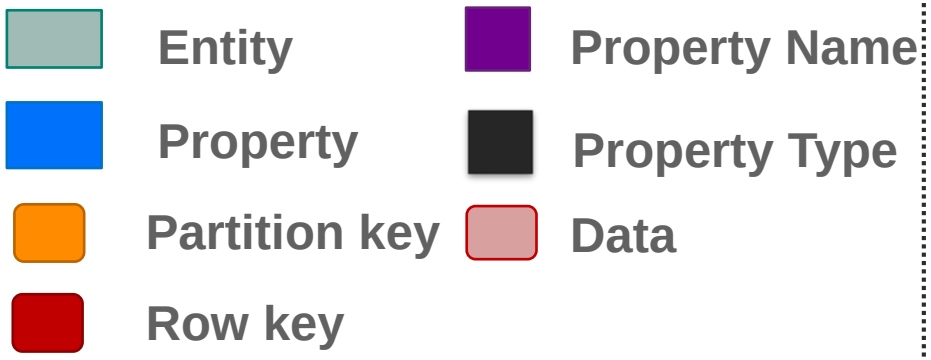
Low cost

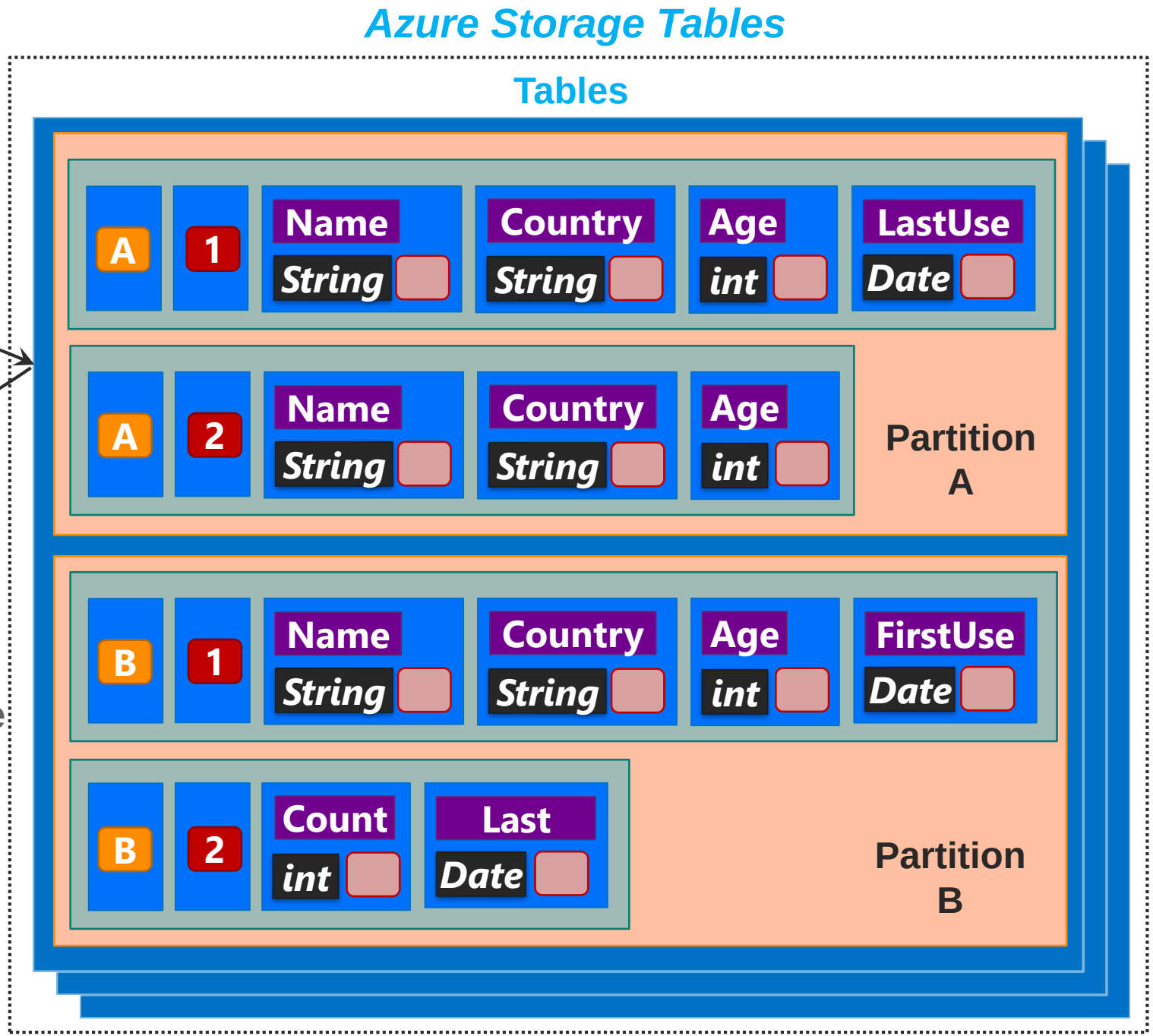
Commonly just a few cents per gigabyte per month

Tables

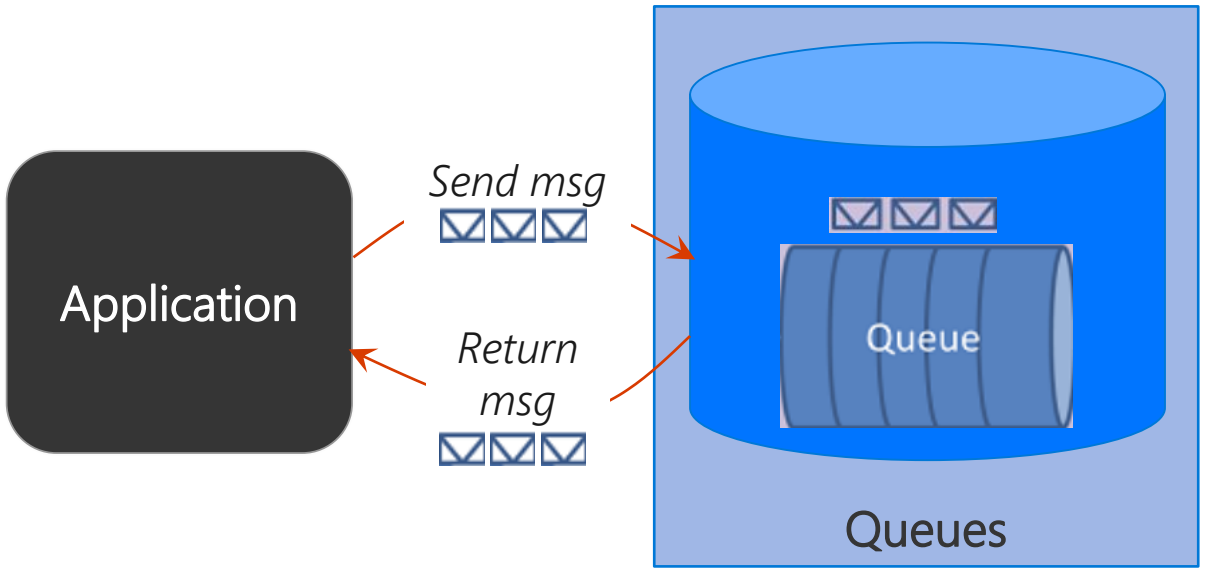
A key/value store





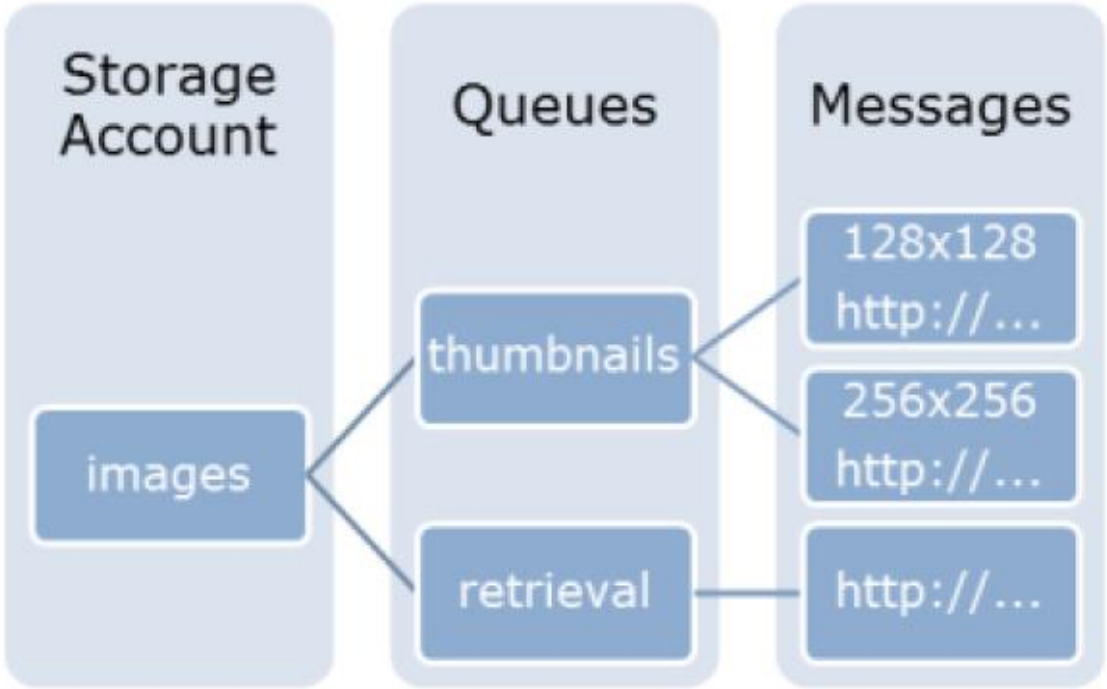


Azure Queues
Reliable messaging system



Queue Storage

- Perfect for asynchronous communication between application components
  - Components are often decoupled (web layer vs business processing layer)
  - Allows scaling components independently
- At-least-once semantic
- Poison message handling
- Max message size: 64 KB, no other limits
- Support for asynchronous tasks
- Can be serviced by Azure Functions
- Queue-Centric Workflow Pattern
- Up to 20000 messages/s



Queue Storage

Example code

C#

```
var storageAccount = CloudStorageAccount.Parse(
                                    "DefaultEndpointsProtocol=https;...");
var queueClient = storageAccount.CreateCloudQueueClient();
var queue = queueClient.GetQueueReference("queue1");
queue.CreateIfNotExists();

var message = new CloudQueueMessage("Hello world from C# client");
queue.AddMessage(message);
```

Why Use Queues?

Resilience

If part of your architecture goes down, messages are buffered, and then naturally picked up by other message processing nodes, which maintains the integrity of your workload

Scale for bursts

Applications absorb unexpected traffic bursts, which prevents servers from being overwhelmed by a sudden flood of requests

Low cost

Additional $0.00036 per 10,000 transactions

Data Protection

- LRS – Locally Redundant Storage
  - 3 copies of data in a single data center
- ZRS – Zone-Redundant Storage
  - 3 copies of data in 2-3 facilities across 1-2 regions
  - Block blobs
- GRS – Geo-Redundant Storage
  - 6 copies of data: 3 copies in primary region, and 3 copies in a secondary region hundreds of miles away
- RA-GRS – Read-access geo-redundant storage
  - GRS with read access to data in secondary location

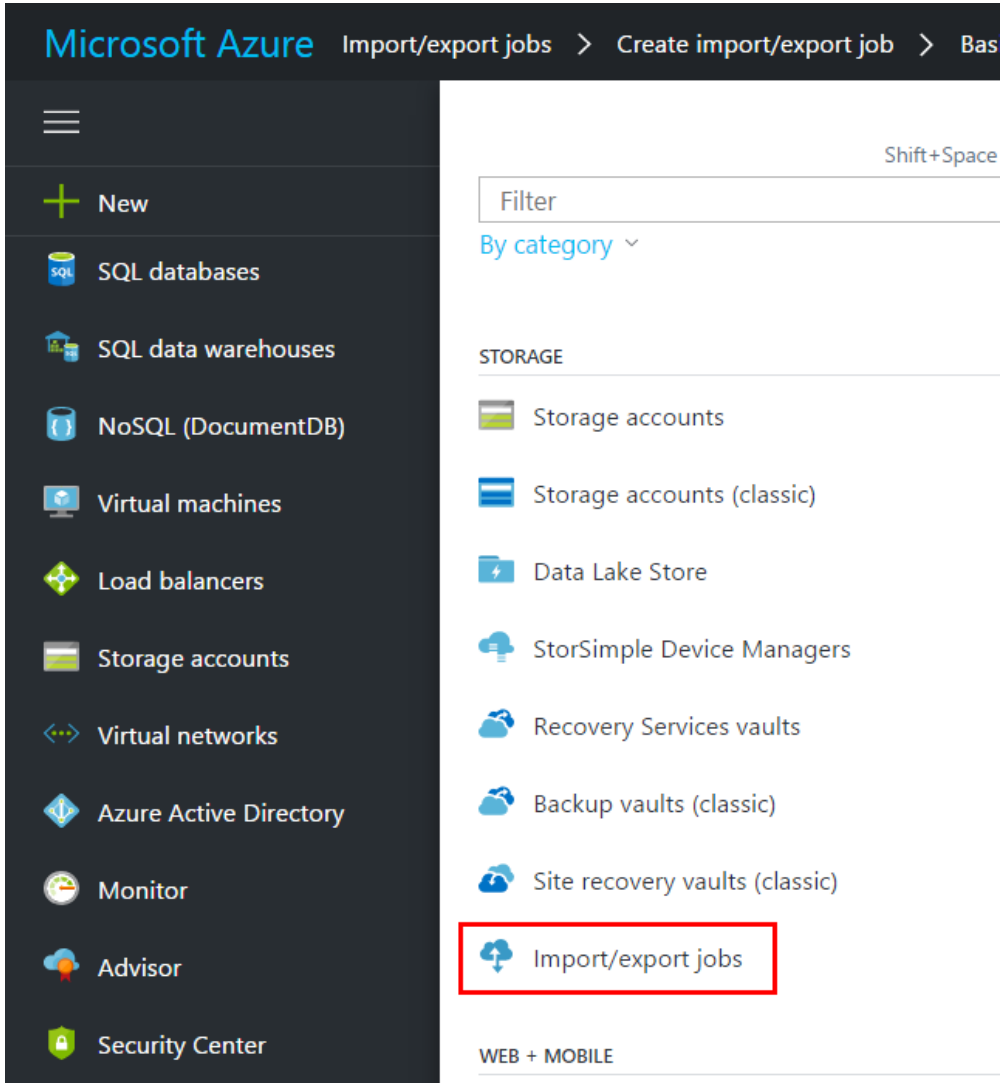
Storage Account Security Features

Security

- Azure Management Plane security
  - Role-based Access Control for storage account
- Azure Data Plane security
  - Storage account keys
  - Storage access policies, shared access signatures (service-level SAS and account-level SAS)
    - Time limited
    - Manage access for specific files, blobs, tables, rows, queues
- Encryption in transit – HTTPS, SMB 3.0
- Encryption at rest
  - Storage Service Encryption
    - Data automatically encrypted prior to persisting, decrypted prior to retrieval
    - Totally transparent, keys managed by Microsoft (for now)
  - Client-side Encryption
    - Built in client libraries
    - Azure Key vault, CEK, KEK
- CORS (Cross-Origin Resource Sharing)

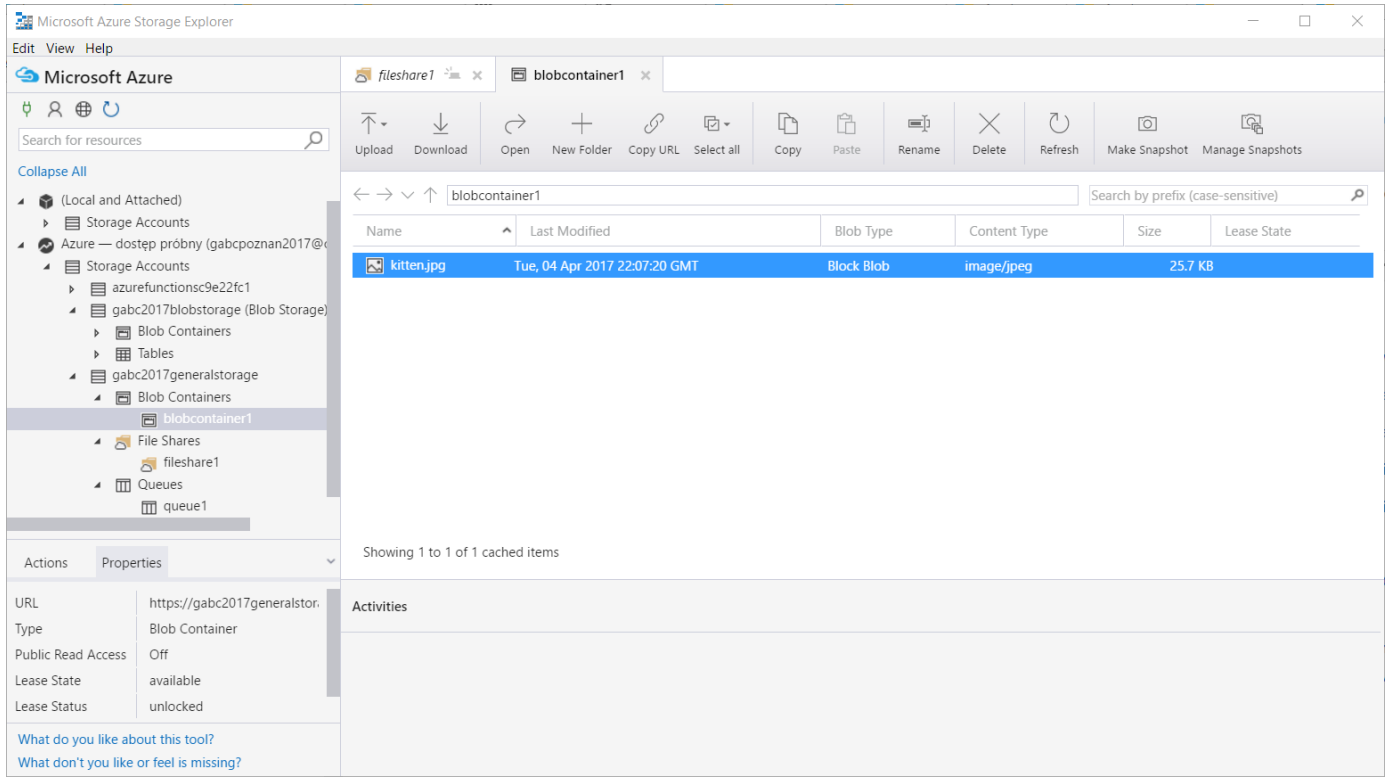
Importing Large Datasets

- It may be infeasible to transfer very large data sets over the wire
- Import/Export service
  - Import Block or Page blobs
  - Export Block, Page and Append blobs
  - Import/Export File Storage
- Prepare the job
  - WAImportExport tool
  - REST API
- Data is encrypted using BitLocker
- Physical hard disk is sent to an Azure data center
- Data is imported into/exported from Azure Storage
- Integrity can be checked using MD5
- Disk is sent back to customer



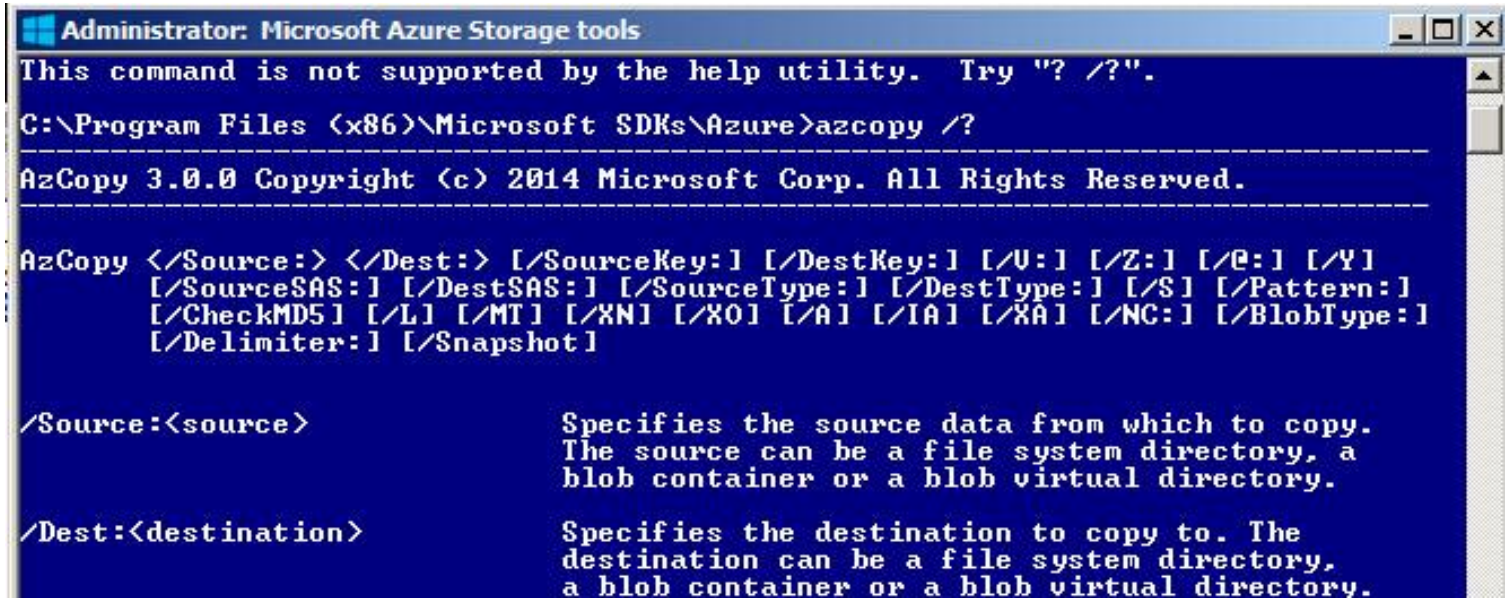
Familiar Tools - Storage Explorer

- Free, standalone, multiplatform (Windows, MacOS, Linux) app for managing storage accounts
- Connects to all kinds of Azure Storage services
- All basic storage operations are covered



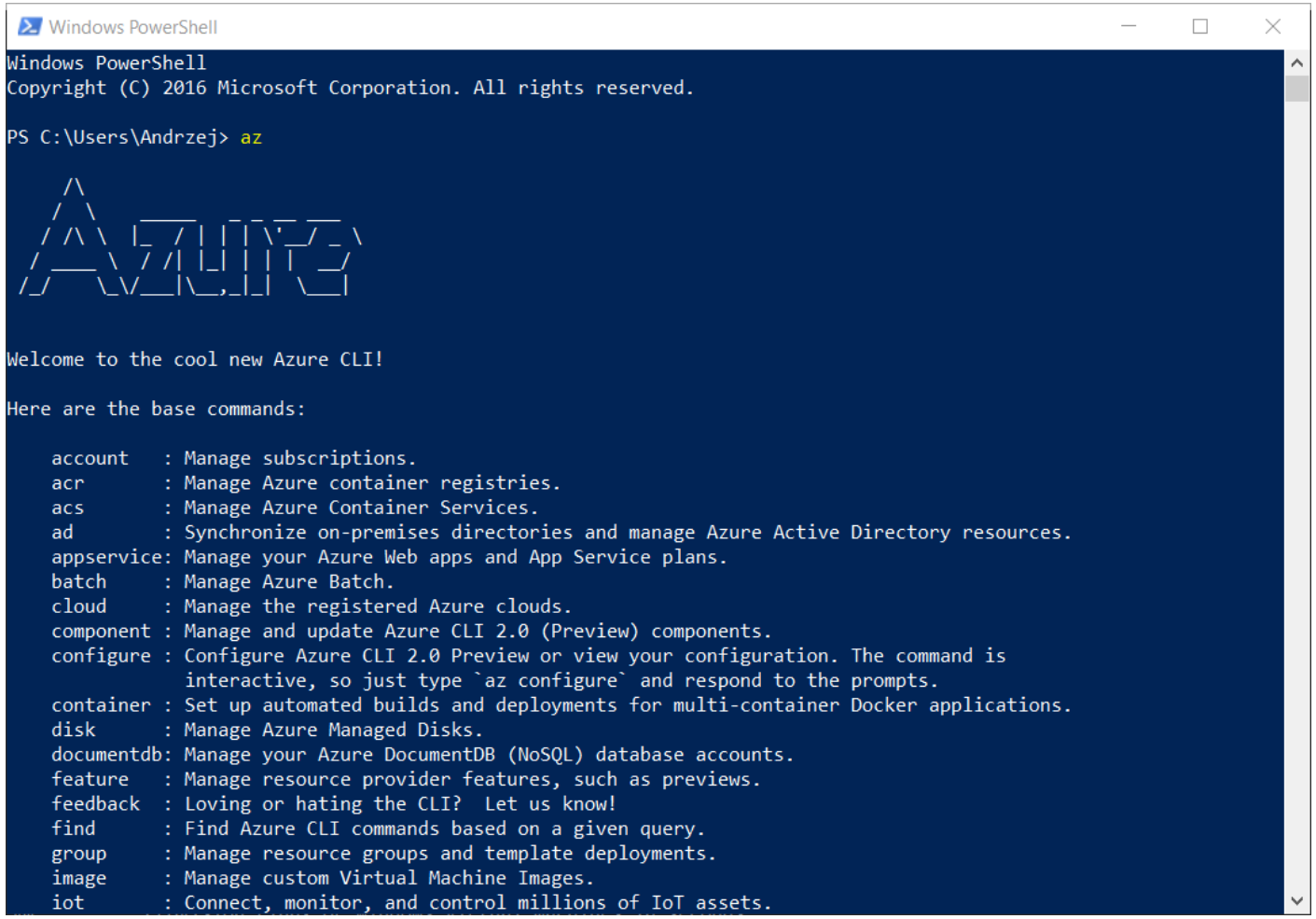
Familiar Tools - AzCopy

- Robust Windows command-line utility
- Copy data to/from Blob/File/Table storage
- Local->Azure, Azure->Local
- Azure->Azure
  - Blob<->Blob, Blob<->File, Blob<->Table
- Lots of options



Familiar Tools - Azure CLI 2.0

- Multi-platform command-line interface for Azure management
- Storage operations (copy, upload, download, delete etc.)
- https://docs.microsoft.com/en-us/azure/virtual-machines/azure-cli-arm-commands#azure-storage-commands-to-manage-your-storage-objects



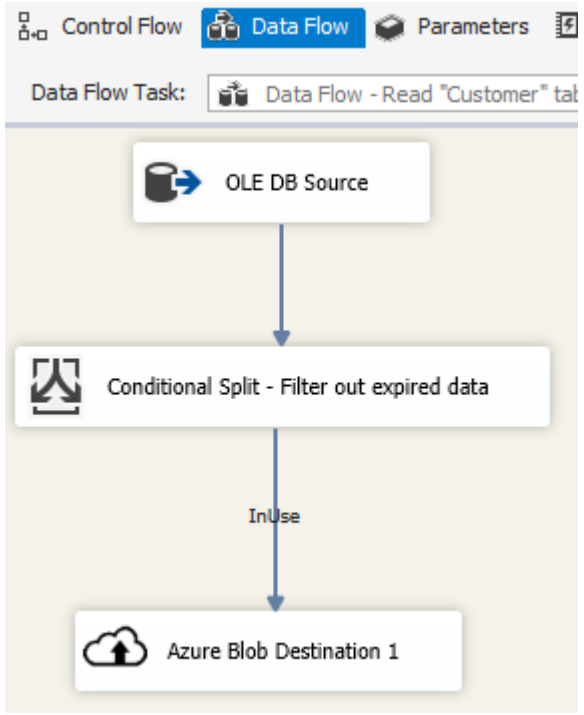
Familiar Tools - PowerShell

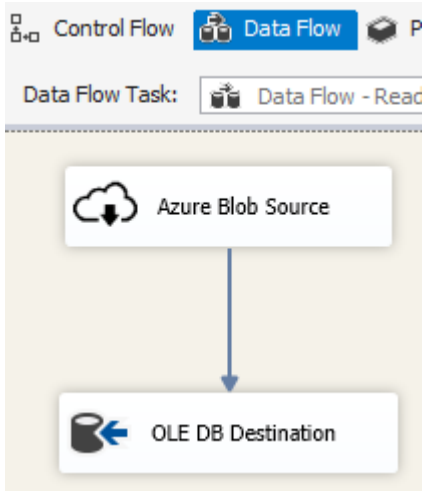
- Modern, .NET based, object-oriented command-line shell
- Full coverage of Azure services
- Resource management
- Resource use

```
1   Import-Module AzureRM
2
3   Login-AzureRmAccount
4
5   $context = New-AzureStorageContext -ConnectionString "DefaultEndpointsF
6
7   # set storage context for the following operations
8   Set-AzureRmCurrentStorageAccount -ResourceGroupName gabc -Name gabc2017
9
10  Get-AzureStorageQueue
11  Get-AzureStorageTable
12
13  Get-AzureStorageBlob -Container blobcontainer1
14  Get-AzureStorageBlobContent -Container blobcontainer1 -Blob kitten.jpg
15
16  Get-AzureStorageFile -ShareName fileshare1
17  Get-AzureStorageFileContent -ShareName fileshare1 -Path kitten.jpg
```

Familiar Tools - Azure Feature Pack for Integration Services

- Blob upload/download task
- Blob source, blob destination
- https://docs.microsoft.com/en-us/sql/integration-services/azure-feature-pack-for-integration-services-ssis





Azure Storage - Summary

- Scalable, durable, highly available data storage
- Can store petabytes of data
- Elastic – it always works the same no matter how much data it contains
- Data accessible from anywhere in the world, from any type of application
- Secure – multiple security layers to keep data private
- Auto-partitioning system – automatically load-balance data based on traffic
- Supports popular languages + REST API for any HTTP/HTTPS client
- Premium Storage – high-perf, low-latency storage for I/O intensive workloads on Azure VMs (backed by SSD disks)
- Easy management and monitoring using Azure Portal or automation

Comparing Azure to on-premises storage

The term "on-premises" refers to the storage and maintenance of data on local hardware and servers.

Cost effectiveness	Reliability	Storage types	Agility
On-premises storage requires up-front expenses. Azure data storage provides a pay-as-you-go pricing model.	Azure data storage provides backup, load balancing, disaster recovery, and data replication to ensure safety and high availability. This capability requires significant investment with on-premises solutions	Azure data storage provides a variety of different storage options including distributed access and tiered storage.	Azure data storage gives you the flexibility to create new services in minutes and allows you to change storage back-ends quickly.

What is serverless?

Full abstraction of servers

Developers can just focus on their code—there are no distractions around server management, capacity planning, or availability

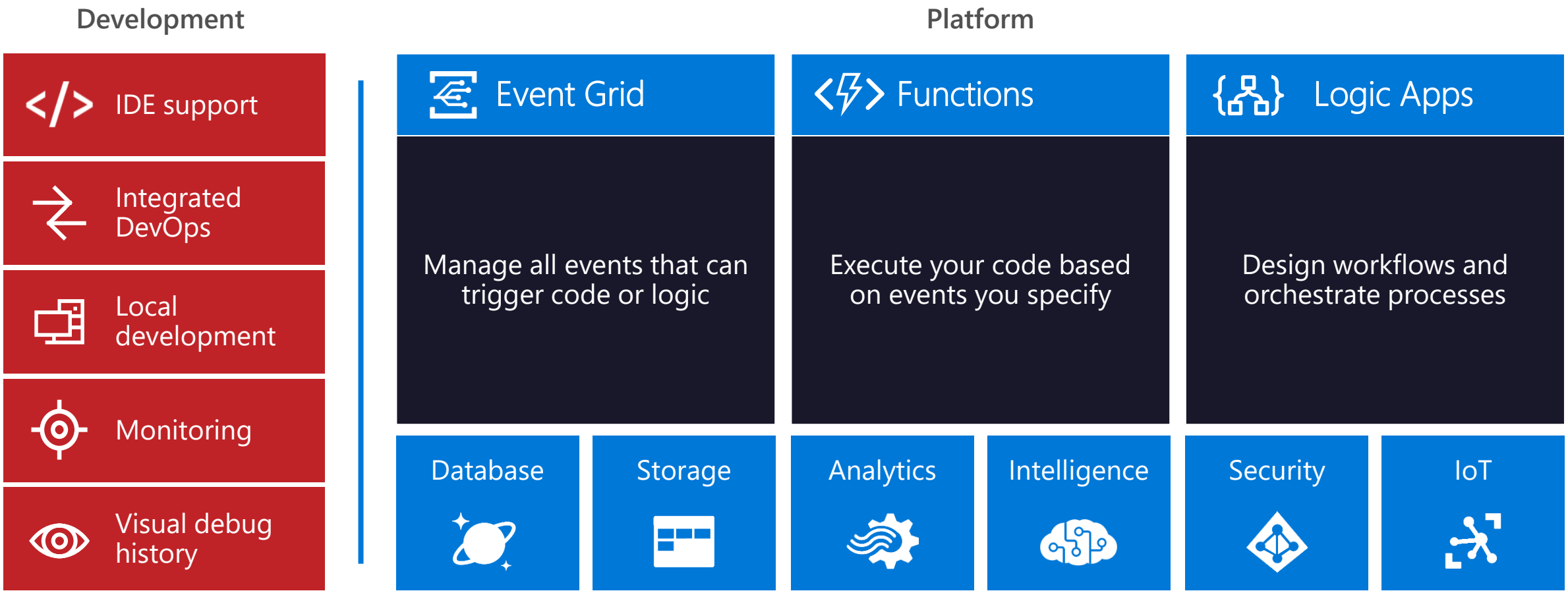
Instant, event-driven scalability

Application components react to events and triggers in near real-time with virtually unlimited scalability; compute resources are used as needed

Pay-per-use

Only pay for what you use: billing is typically calculated on the number of function calls, code execution time, and memory used

Azure Serverless Ecosystem



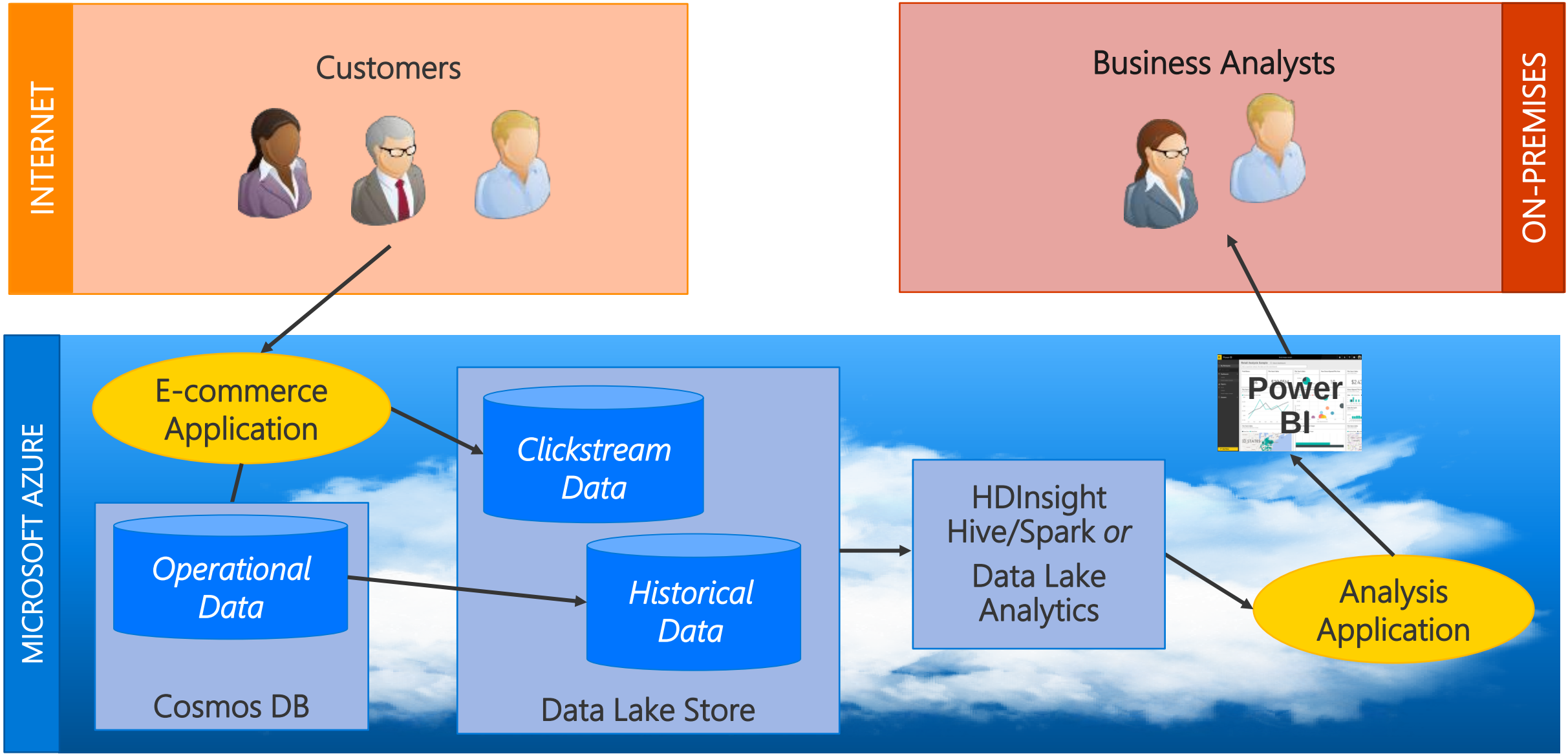
Lab Custom Vision, Storage and Logic Apps

- Resource provisioning
- Create Custom Vision project
- Build Custom AI into an Application using Azure Logic Apps

Data Lake Storage

Analyzing Unstructured Data

A scenario



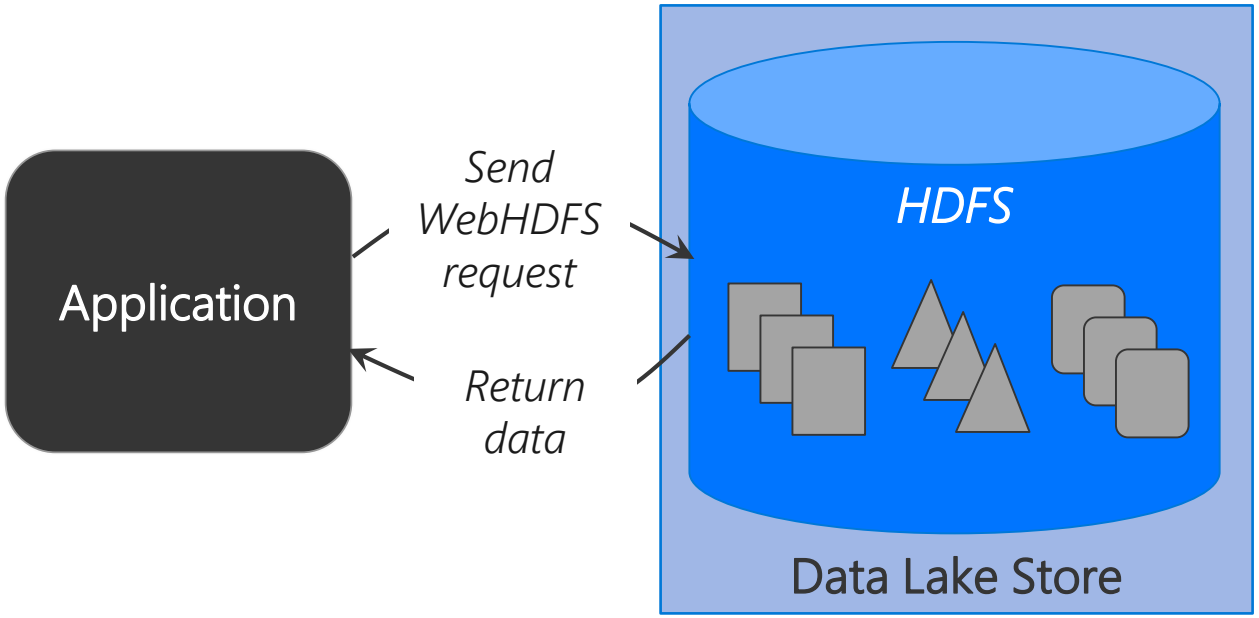
Data Lakes

If you think of a datamart as a store of bottled water - cleansed and packaged and structured for easy consumption - the data lake is a large body of water in a more natural state. The contents of the lake stream in from a source to fill the lake, and various users of the lake can come to examine, dive in, or take samples.

James Dixon CEO Pentaho

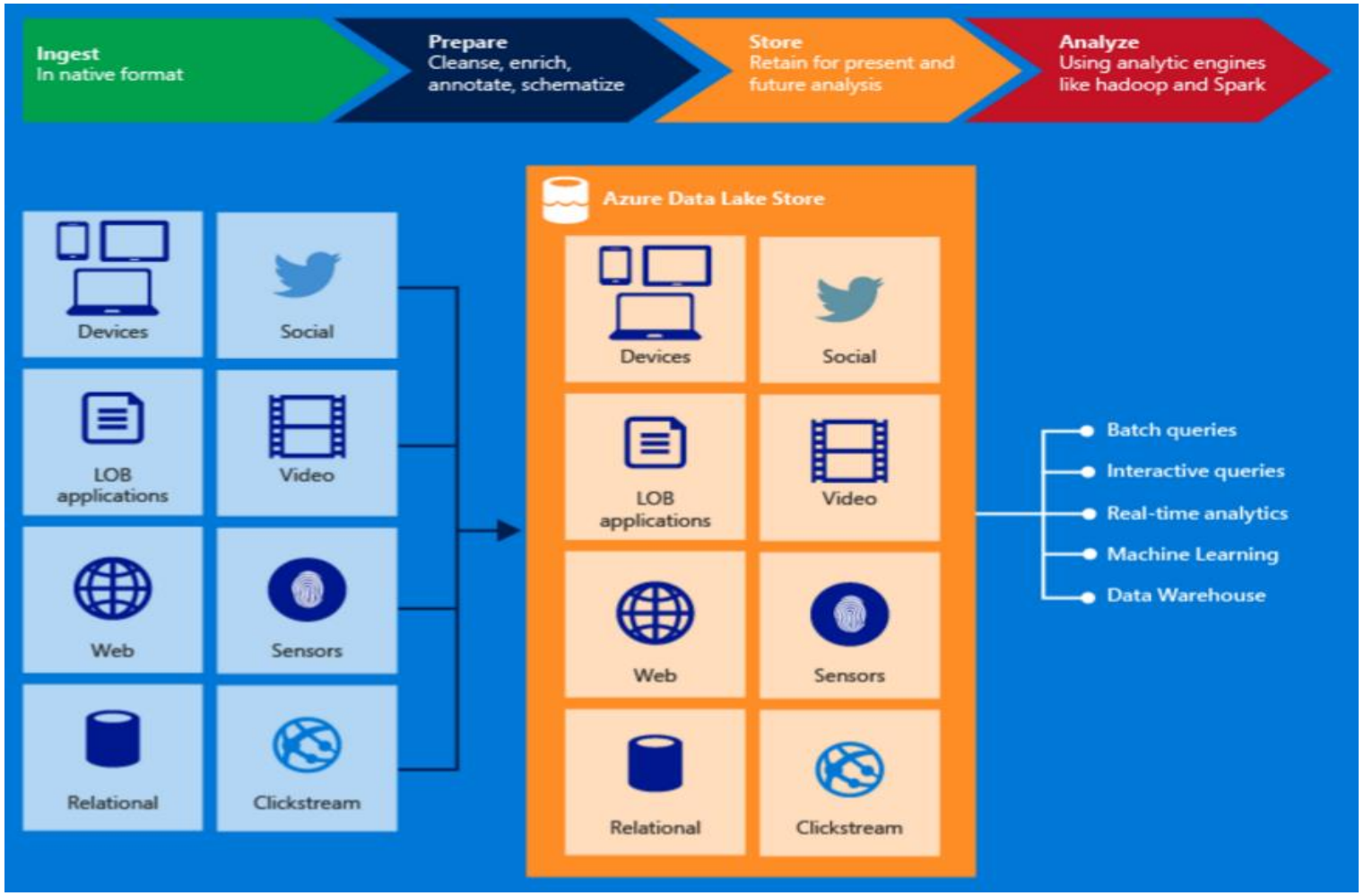
Data Lake Store

An illustration

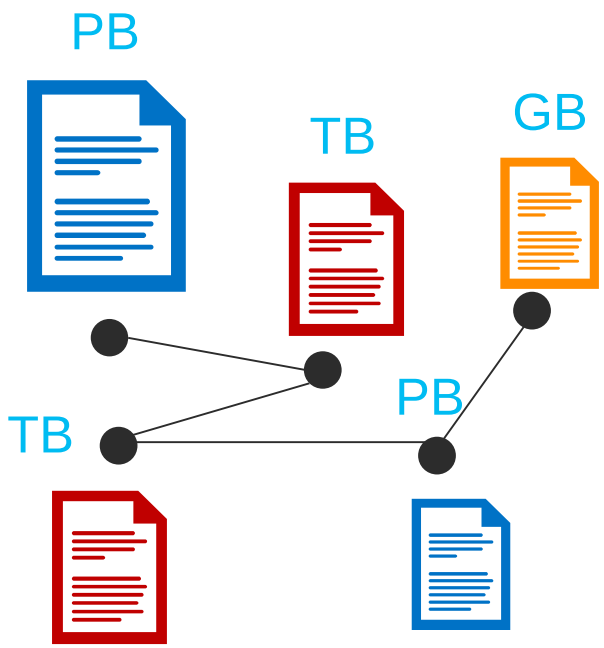


Azure Data Lake Store

- Enterprise-wide hyper-scale repository for big data analytic workloads
- Enables you to capture data of any size, type, and ingestion speed in one single place for operational and exploratory analytics



Data stored of any size, optimized for high performance



Store has no fixed limits on file sizes
(PB sized files)

Ultra fast read/write access

No code rewrite as you increase size of data stored

Optimized for large analytic systems: with massive throughput

Optimized for IOT with high volume of small writes

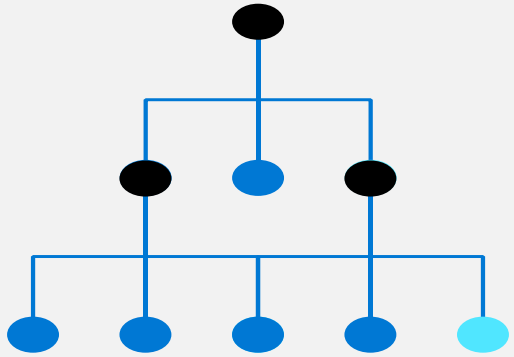
Useful for very large data or for real-time

Azure Data Lake Storage – Generation II

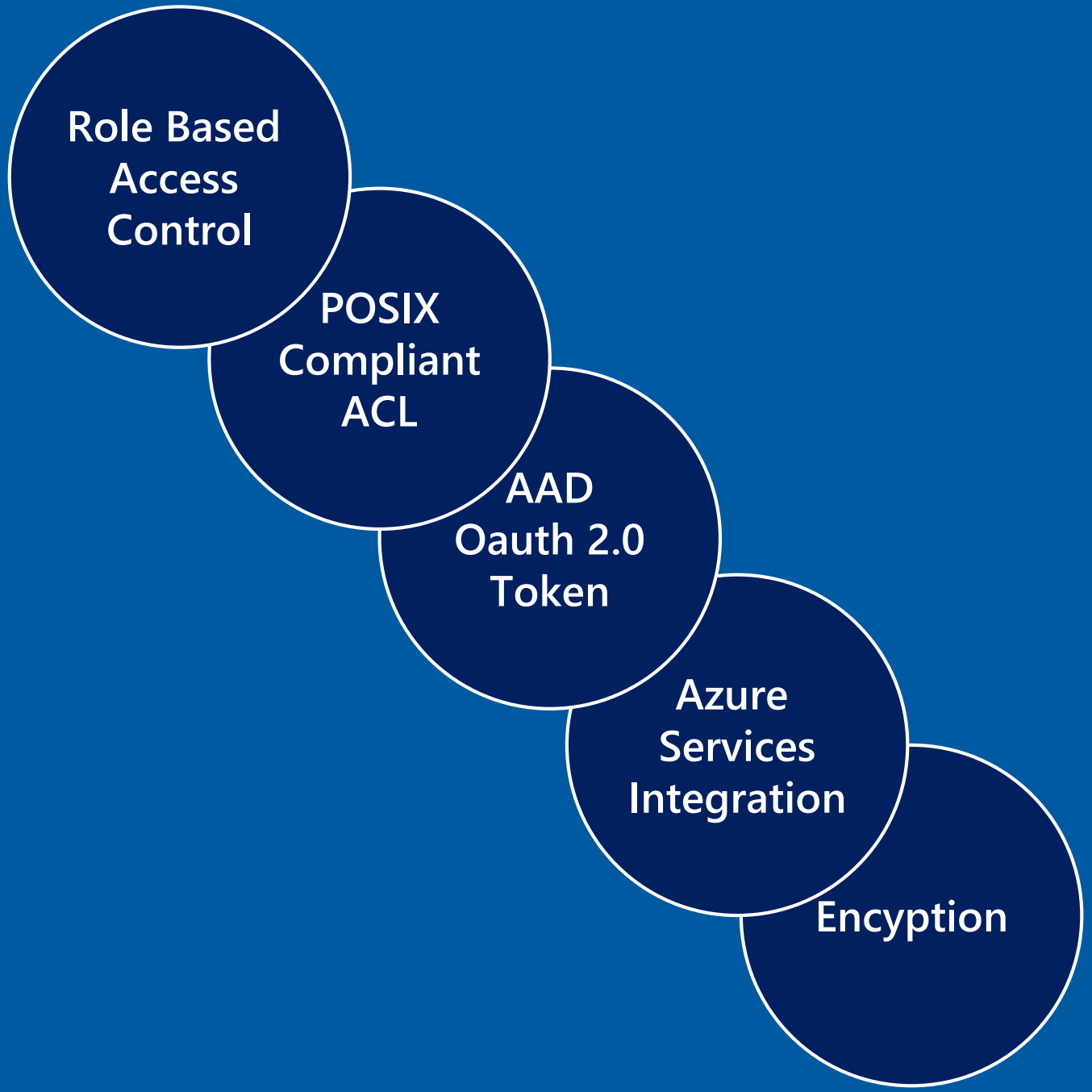
Compare Azure Blob Storage and Data Lake Store Gen 2

Azure Blob
Flat Namespace

Data Lake (Gen II)
Hierarchical Namespace



Azure Data Lake Storage Gen2 Security Features



Why use Data Lake Store?

Provides HDFS as a service

HDFS has become the default storage for data lakes

Exposes a standard API

The WebHDFS API is an Apache project

Optimized for analytic workloads

Compared to blobs:

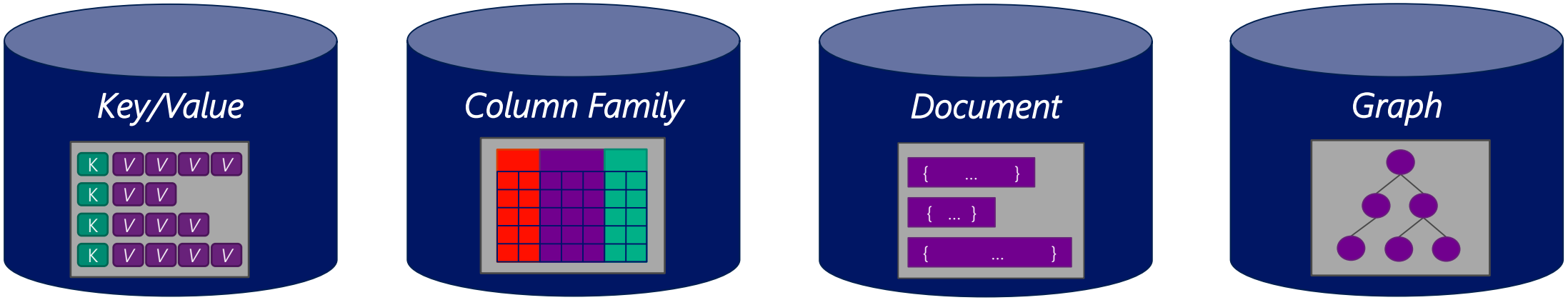
- Designed to be accessed by parallel applications

- A single HDFS file can be as large as a petabyte

Cosmos DB

NoSQL Data Models

Leading approaches



Why Use a NoSQL Service for Operational Data?

To scale for lots of users and lots of data

Pros: NoSQL technologies can offer more scalability than relational databases

Cons: Often lose some benefits of relational databases, e.g., database-wide transactions

To work with data in a more flexible way

Pros: NoSQL technologies don't have fixed schemas

Cons: Fixed schemas help prevent errors

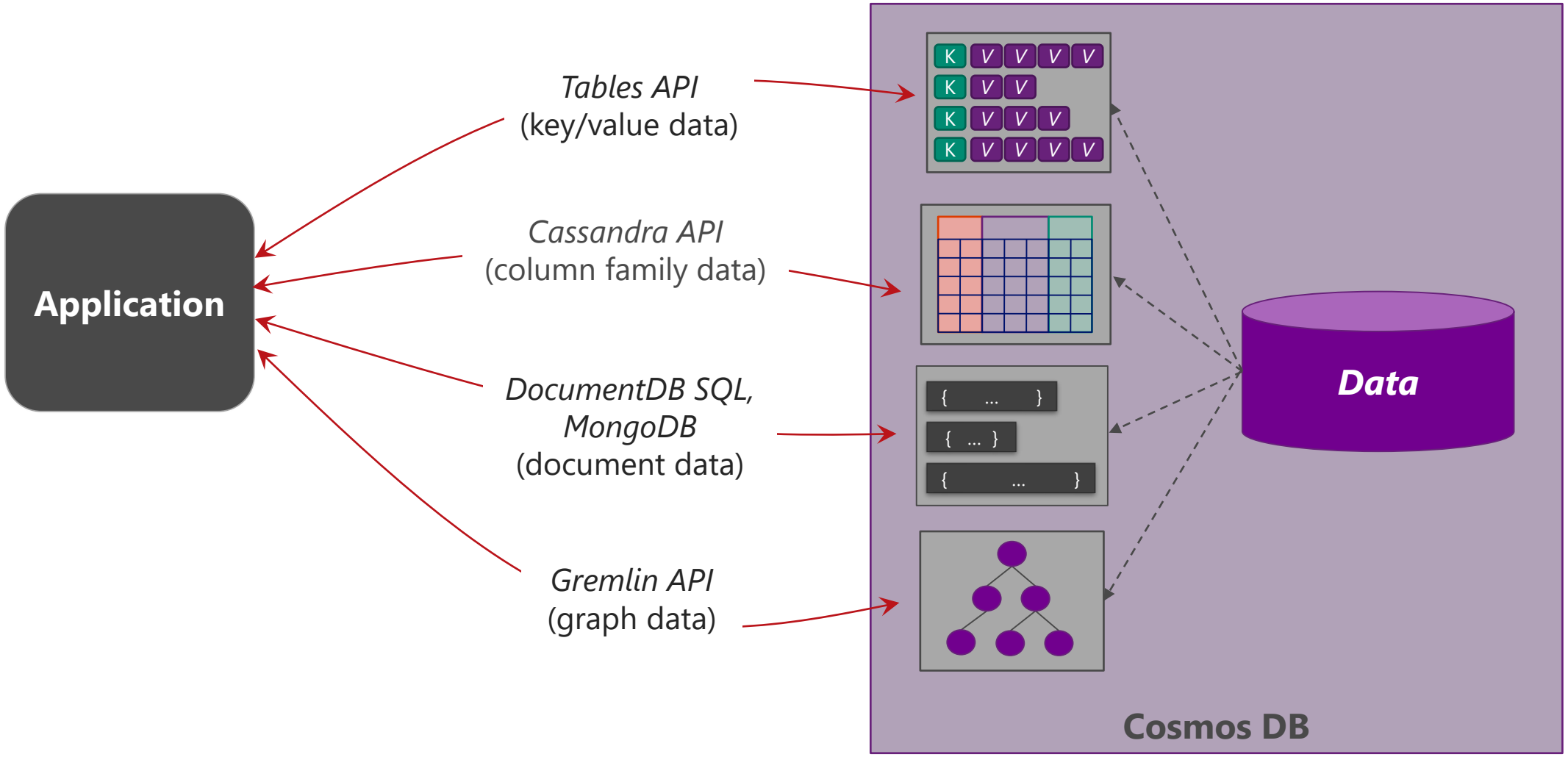
To work better with different data models

Pros: Non-relational data is today's growth area

Cons: The data models are less familiar

Cosmos DB

A multi-model store



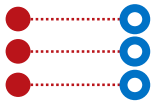
Azure Cosmos DB

A globally distributed, massively scalable, multi-model database service

Table API

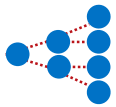
SQL

MongoDB API

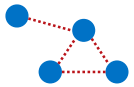


Key-value

Column-family



Document



Graph

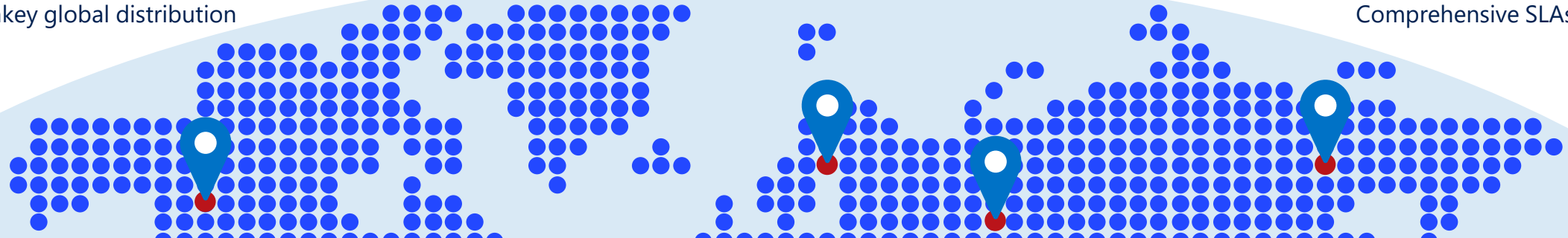
Guaranteed low latency at the 99th percentile

Elastic scale out
of storage & throughput

Five well-defined consistency models

Turnkey global distribution

Comprehensive SLAs



Global Distribution From The Ground-Up

- Azure Cosmos DB is a Foundational (Ring 0) Azure service
  - Available in all Azure regions by default, including sovereign/government clouds
- Transparent and automatic multi-region replication
  - Associate any number of regions with your database account, at any time
  - Policy based geo-fencing
- Multi-homing APIs
  - All endpoints are logical, by default
  - Apps don’t need to be redeployed during regional failover
  - Apps can also access physical endpoints if needed
- Support for both manual and automatic failover
- Designed for high availability
  - Allows for dynamically setting priorities to regions
  - Simulate regional disasters via API
  - Test the end-to-end availability for the entire app (beyond just the database)
- Comprehensive SLAs
  - First and only to offer comprehensive SLA for latency, throughput, availability and consistency

Guaranteed low latency – Value to Customer

Reads and writes served from local region

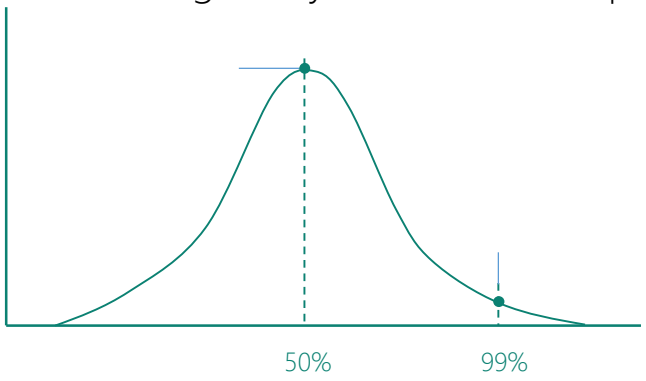
Guaranteed millisecond latency worldwide

Write optimized, latch-free database engine

Automatically indexed SSD storage

Synchronous and automatic indexing at sustained ingestion rates

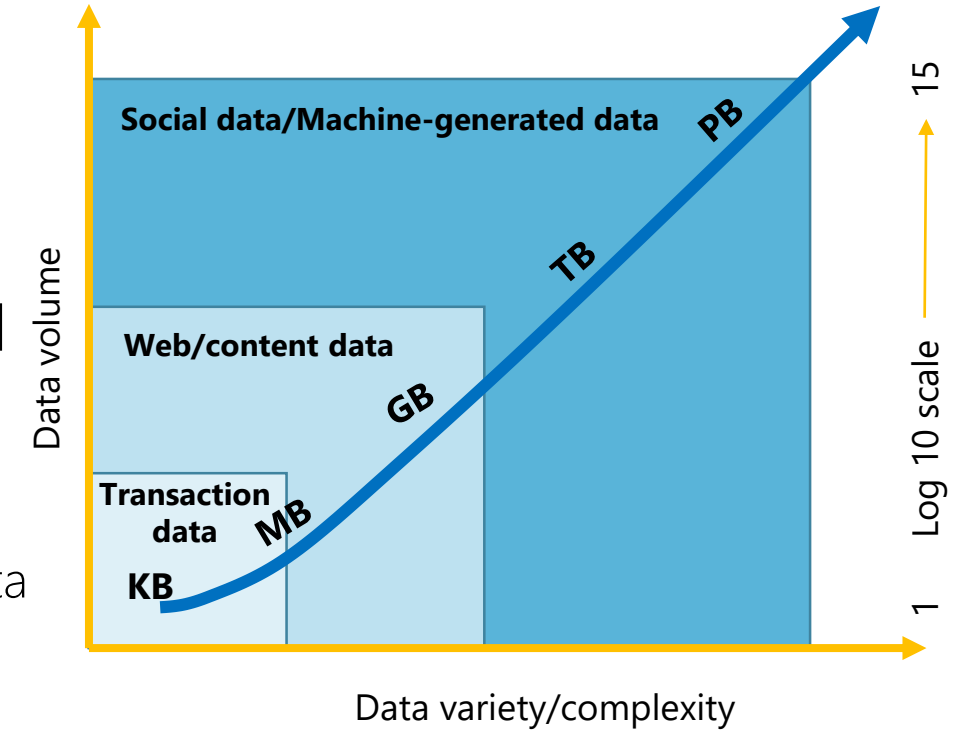
- No schema or index management needed
- No schema versioning needed
- No schema migration needed
- All of this is highly relevant for rapidly evolving apps in a globally distributed setup



	Reads (1KB)	Indexed writes (1KB)
50th	<2ms	<6ms
99th	<10ms	<15ms

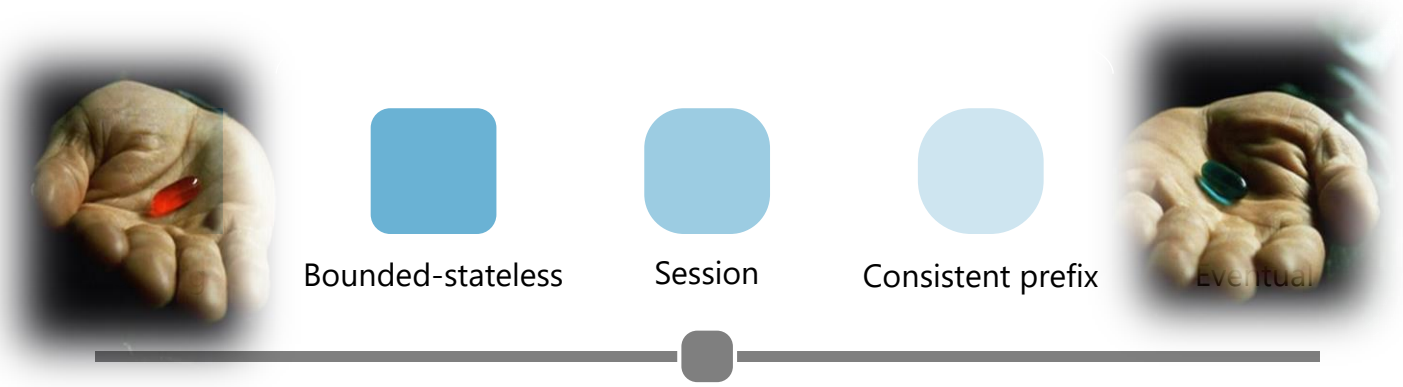
Azure Cosmos DB: Elastically Scalable Storage

- Single machine is never a bottleneck
- A single table can scale from GB-PBs, across many machines, and regions
- Transparent server side partition management and routing
- Optionally evict old data using built-in support for TTL
  - Policy based, automatic tiering to any HDFS compatible data lake (e.g. ADLS or Azure Storage)
  -
- Customers pay only for the throughput and storage they need



Choices of Consistency

Most real-life applications do not fall into these two extremes



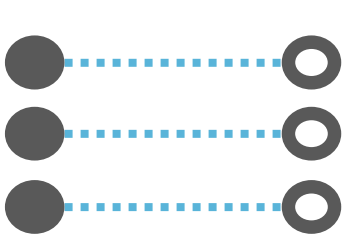
5 well-defined consistency levels for low latency and high availability

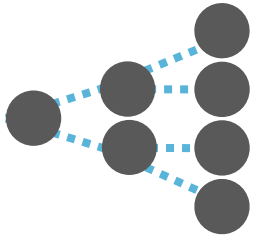
Schema-agnostic, automatic indexing

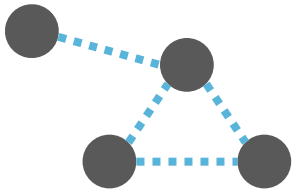
- At global scale, schema/index management is hard
- Automatic and synchronous indexing of all ingested content - hash, range, geo-spatial, and columnar
  - No schemas or secondary indices ever needed
- Resource governed, write optimized database engine with latch free and log structured techniques
- Online and in-situ index transformations
- While the database is fully schema-agnostic, schema-extraction is built in
  - Customers can get Avro schemas from the database

Native Support for Multiple Data Models

- Database engine operates on atom-record-sequence (ARS) based type system
  - All data models are **efficiently** translated to ARS
- API and wire protocols are supported via extensible modules
- Instance of a given data model can be materialized as trees
- Graph, documents, key-value, column-family, … *more to come*







Who Wants to Have 3-5 Different Backend Databases?

Security & Compliance

- Enterprise grade security
- Encryption at Rest
  - Always encrypted at rest and in motion
  - Data, index, backups, and attachments encrypted
- Encryption is enabled automatically by default
  - No impact on performance, throughput or availability
  - Transparent to your application
- Comprehensive Azure compliance certification
  - ISO 27001, ISO 27018, EUMC, HIPAA, PCI
  - SOC1 and SOC2 (Audit complete, Certification in Q2 2017)
  - FedRAMP, IRS 1075, UK Official (IL2) (Q2 2017)
  - HITRUST (H2 2017)

What are Request Units

Throughput is important to ensure you can handle the volume of transactions you need.

Database Throughput

Database throughput is the number of reads and writes that your database can perform in a single second

What is a Request Unit

Azure Cosmos DB measures throughput using something called a request unit (RU). Request unit usage is measured per second, so the unit of measure is request units per second (RU/s). You must reserve the number of RU/s you want Azure Cosmos DB to provision in advance.

Exceeding throughput limits

If you don't reserve enough request units, and you attempt to read or write more data than your provisioned throughput allows, your request will be rate-limited.

Choosing a Partition Key

Why have a Partition Strategy

Having a partition strategy ensures that when your database needs to grow, it can do so easily and continue to perform efficient queries and transactions
.

What is a Partition Key.

A partition key is the value by which Azure organizes your data into logical divisions.

Best Practice.

Range of Values

The more values your partition key has, the more scalability you have.

Review Queries

To determine the best partition key for a read-heavy workload, review the top three to five queries you plan on using.

Transactional Workloads

For write-heavy workloads, you'll need to understand the transactional needs of your workload

Running complex operations on data

Multiple documents in your database frequently need to be updated at the same time. The way to perform these transactions in Azure Cosmos DB is by using stored procedures and user-defined functions (UDFs).

Stored Procedures

Stored procedures perform complex transactions on documents and properties. Stored procedures are written in JavaScript and are stored in a collection on Azure Cosmos DB.

User Defined Functions

User Defined Functions are used to extend the Azure Cosmos DB SQL query language grammar and implement custom business logic, such as calculations on properties and documents.

Why use Cosmos DB?

Globally distributed

With low-latency access from anywhere in the world

Multiple consistency options

Strong

Eventual

Session

More

99% SLAs for many things

Availability

Throughput

Latency (<10ms reads)

Consistency

Data redundancy

Data redundancy is the process of storing data in multiple locations to ensure that it is highly available.

Azure Blob and Data Lake Store	SQL Data Warehouse	Cosmos DB	SQL Database
• Locally redundant storage (LRS) • Zone-redundant storage (ZRS) • Geo-redundant storage (GRS) • Read-access geo-redundant storage (RA-GRS)	SQL Data Warehouse performs a **geo-backup** once per day to a paired data center. The RPO for a geo-restore is 24 hours.	Azure Cosmos DB is a globally distributed database service. You can configure your databases to be globally distributed and available in any of the Azure regions.	• Check that you have provisioned enough DTU's • Review whether the database would benefit from elastic pools • A wide range of tools can be used to troubleshoot SQL Database

Disaster Recovery

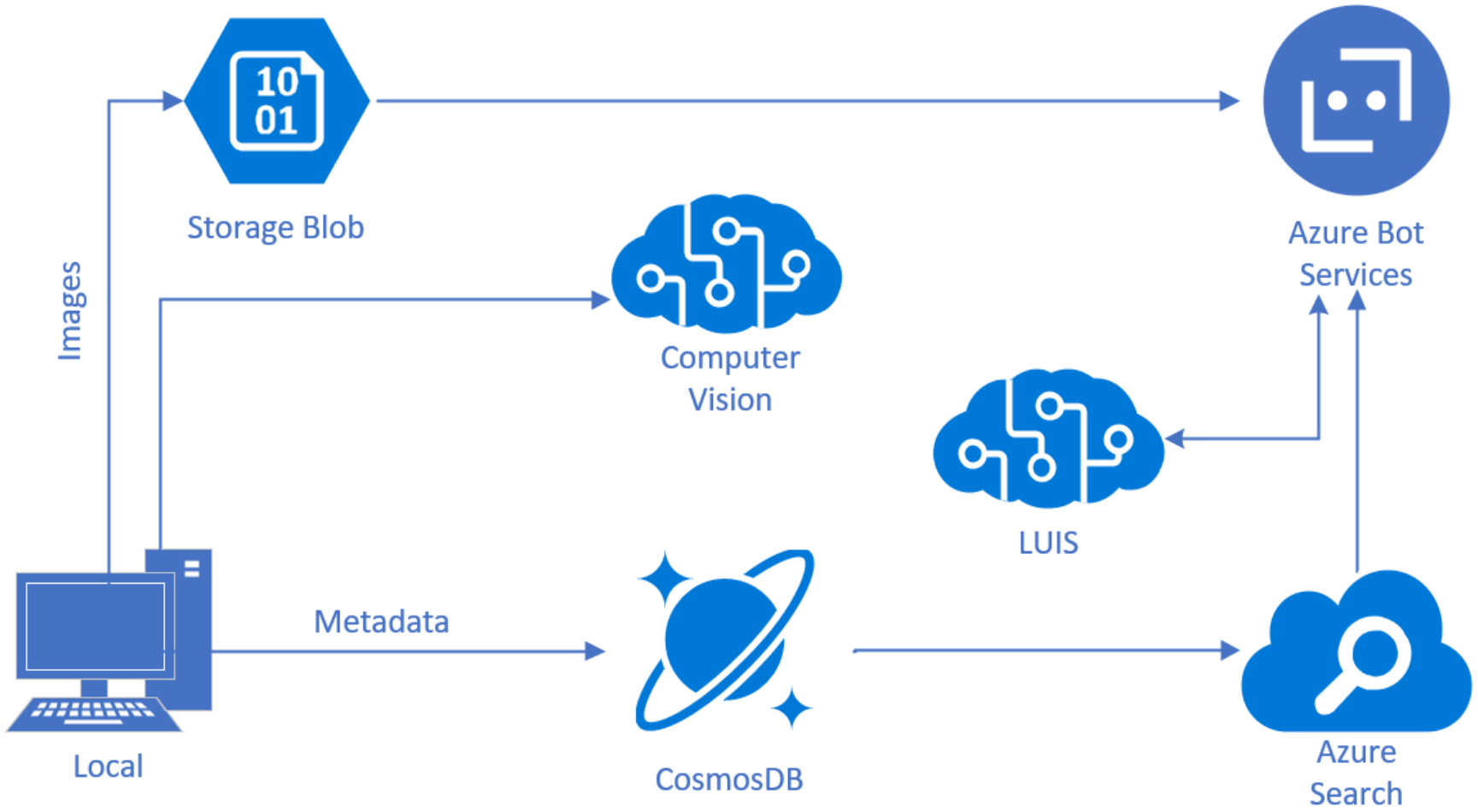
There should be processes that are involved in backing up or providing failover for databases in an Azure data platform technology. Depending on circumstances, there are numerous approaches that can be adopted.

Azure Blob and Data Lake Store	SQL Data Warehouse	Cosmos DB	SQL Database
Supports account failover for geo-redundant storage accounts. You can initiate the failover process for your storage account if the primary endpoint becomes unavailable. .	SQL Data Warehouse performs a **geo-backup** once per day to a paired data center. Data warehouse snapshot feature that enables you to create a restore point to create a copy of the warehouse to a previous state.	Takes a backup of your database every **4 hours** and at any point of time Only the latest 2 backups are stored.	Creates database backups that are kept between 7 and 35 days Uses Azure read-access geo-redundant storage (RA-GRS) to ensure that they preserved even if data center is unavailable.

Intelligent Search Solution for Images

- Now we will focus on hands-on activities that develop proficiency in Azure Cognitive Services
- We're going to build an end-to-end scenario that allows you to pull in your own pictures, use Cognitive Services to find objects and people in the images, obtain a description and tags, and store all of that data into a NoSQL Store (CosmosDB)
- We'll use that NoSQL Store to populate an Azure Search index

Solution architecture



Lab Computer Vision

- Set up Azure resources
- Collect the Keys
- Image Processing

Questions

- Machine Learning practitioner
- Over 25 years of professional experience
- Artificial Intelligence MVP & MCT
- Microsoft Certified Solutions Expert
  - Data Management and Analytics
  - Cloud Platform and Infrastructure
  - Business Intelligence
- Microsoft Certified Solutions Developer
  - Azure Solution Architect