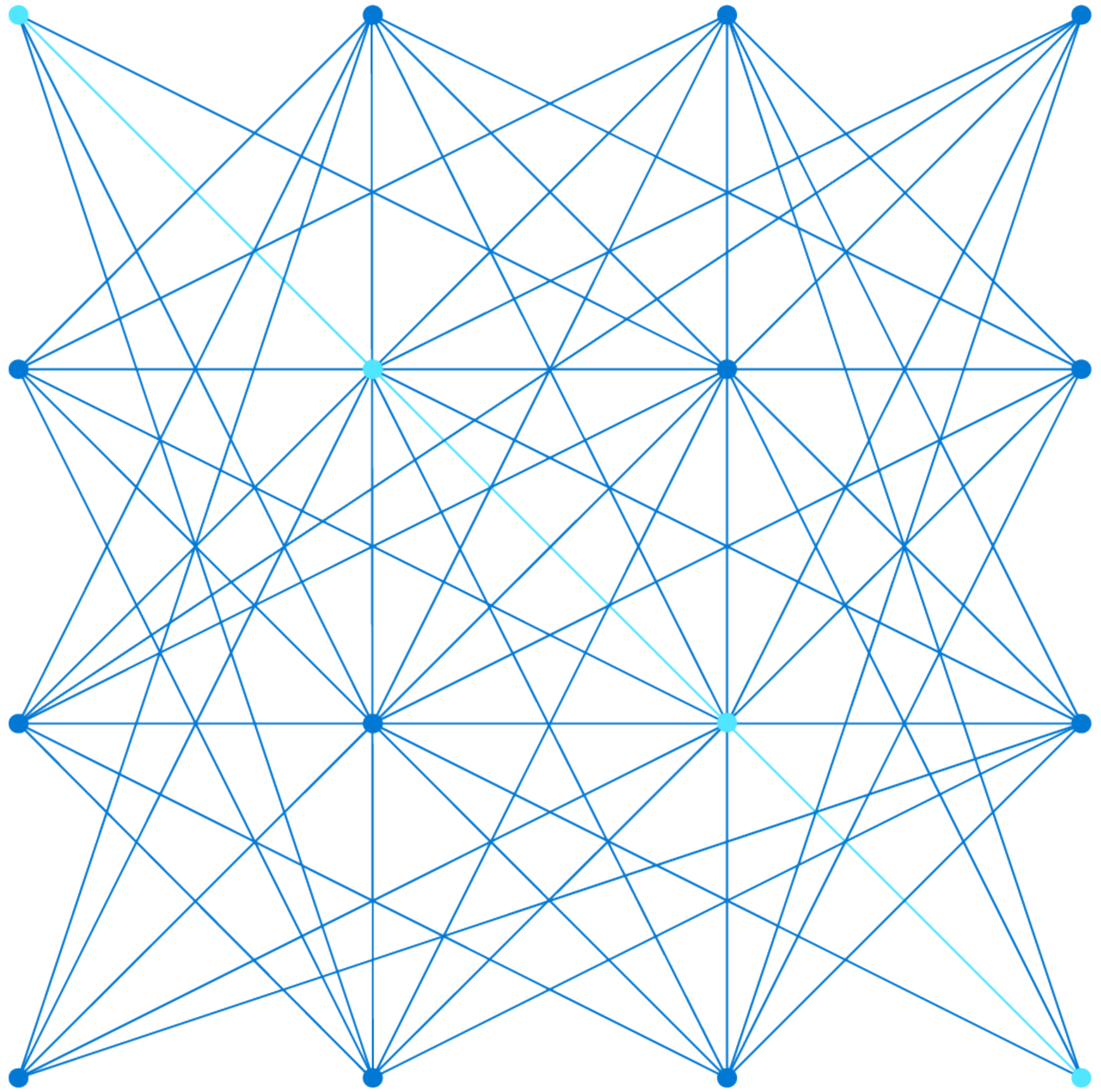
Microsoft Azure

3: Explore fundamentals of non-relational data in Azure



Agenda

Fundamentals of Azure Storage

Fundamentals of Azure Cosmos DB

1: Fundamentals of Azure Storage

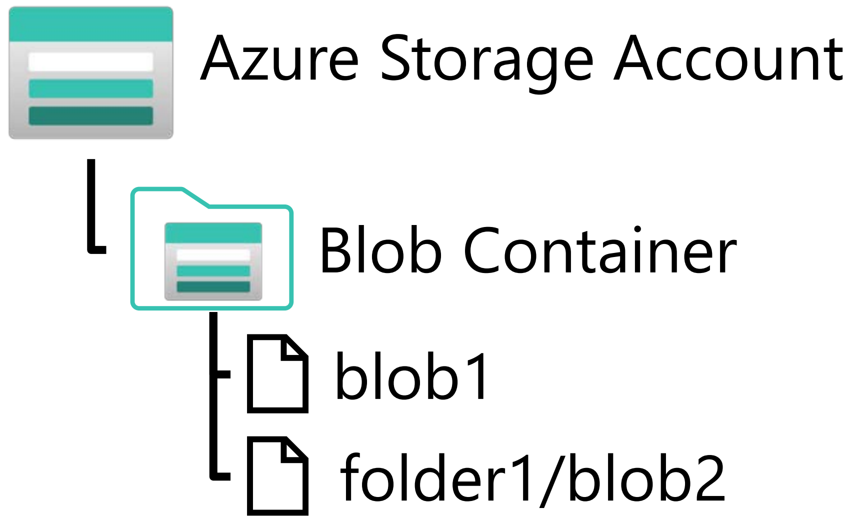
Azure Blob Storage

Storage for data as binary large objects (BLOBs)

- Block blobs
  - Large, discrete, binary objects that change infrequently
  - Blobs can be up to 4.7 TB, composed of blocks of up to 100 MB
    - A blob can contain up to 50,000 blocks
- Page blobs
  - Used as virtual disk storage for VMs
  - Blobs can be up to 8 TB, composed of fixed sized-512 byte pages
- Append blobs
  - Block blobs that are used to optimize append operations
  - Maximum size just over 195 GB - each block can be up to 4 MB

Per-blob storage tiers

- Hot – Highest cost, lowest latency
- Cool – Lower cost, higher latency
- Archive – Lowest cost, highest latency



Blobs can be organized in virtual directories, but each path is considered a single blob in a flat namespace – folder level operations are not supported

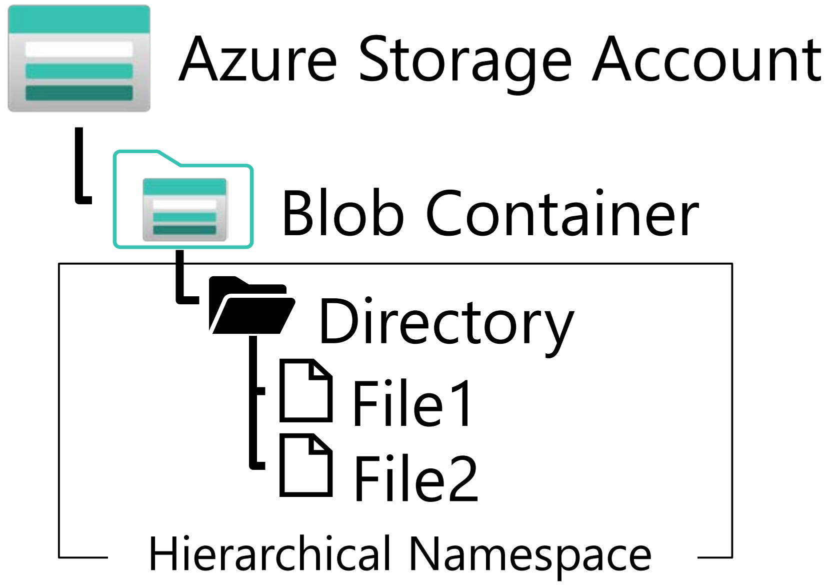
Azure Data Lake Store Gen 2

Distributed file system built on Blob Storage

- Combines Azure Data Lake Store Gen 1 with Azure Blob Storage for large-scale file storage and analytics
- Enables file and directory level access control and management
- Compatible with common large scale analytical systems

Enabled in an Azure Storage account through the *Hierarchical Namespace* option

- Set during account creation
- Upgrade existing storage account
  - One-way upgrade process

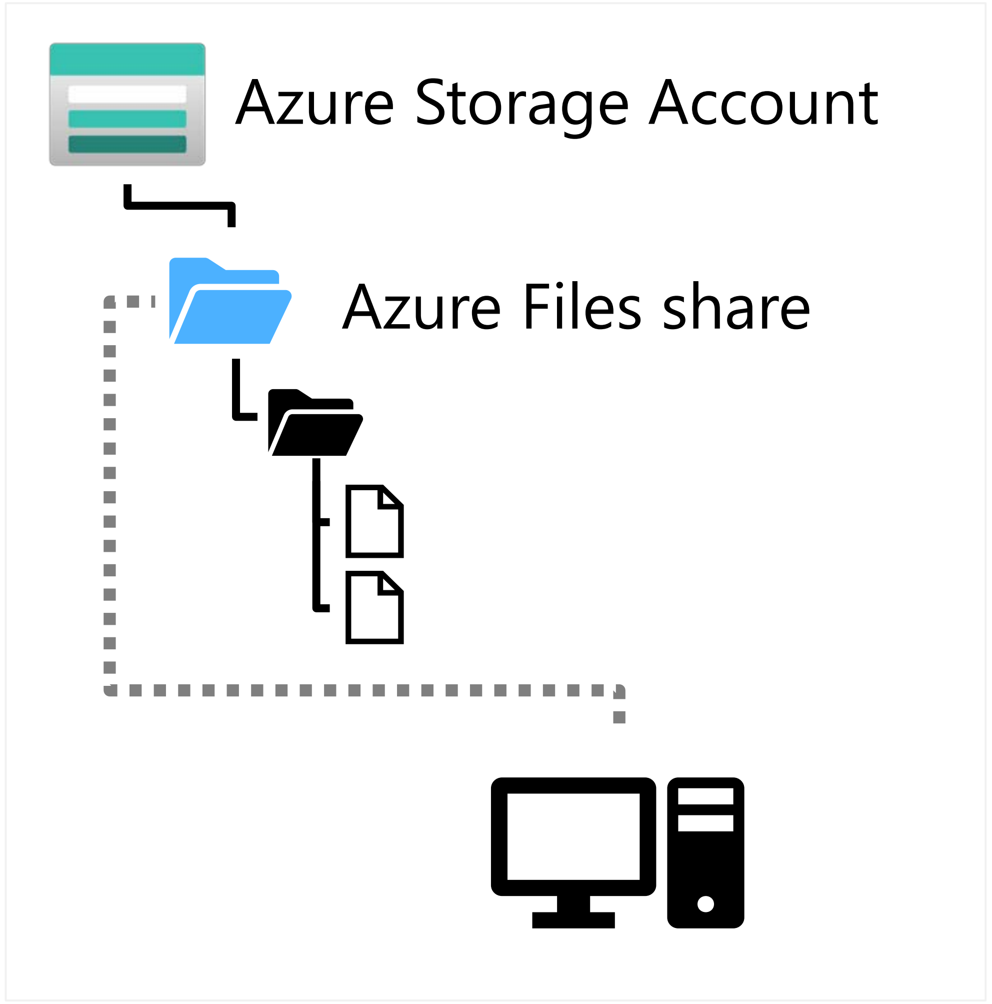


File system includes directories and files, and is compatible with large scale data analytics systems like Hadoop, Databricks, and Azure Synapse Analytics

Azure Files

Files shares in the cloud that can be accessed from anywhere with an internet connection

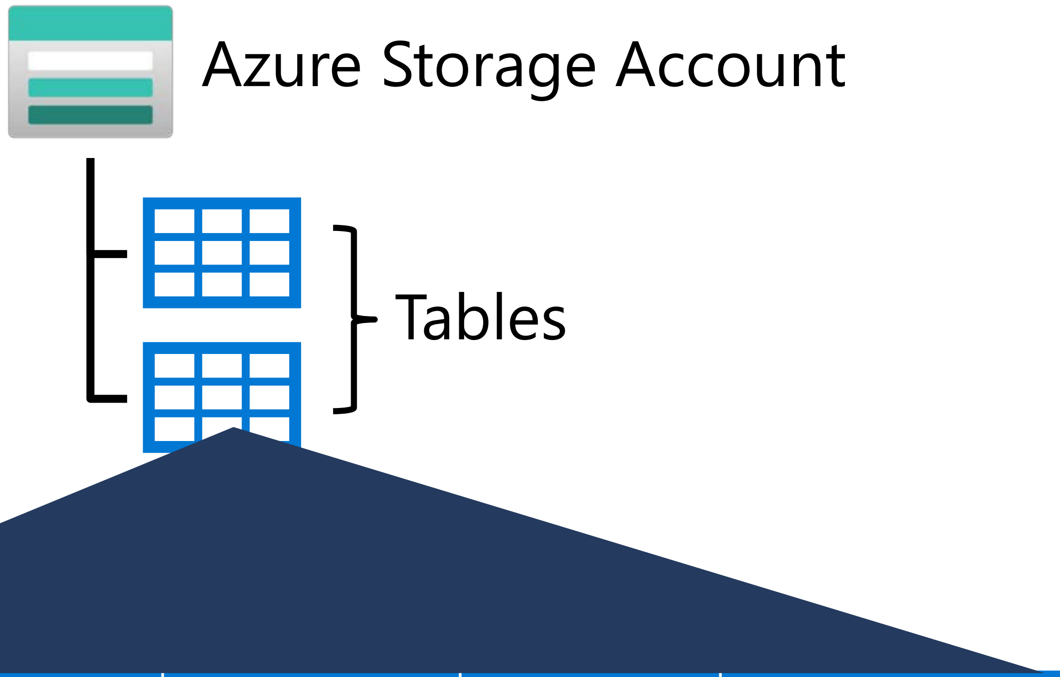
- Support for common file sharing protocols:
  - Server Message Block (SMB)
  - Network File System (NFS) – *requires premium tier*
- Data is replicated for redundancy and encrypted at rest



Azure Table Storage

***Key-Value* storage for application data**

- Tables consist of *key* and *value* columns
  - Partition and row keys
  - Custom property columns for data values
    - A *Timestamp* column is added automatically to log data changes
- Rows are grouped into partitions to improve performance
- Property columns are assigned a data type, and can contain any value of that type
- Rows do not need to include the same property columns



PartitionKey	RowKey	Timestamp	Property1	Property2
1	123	2022-01-01	A value	Another value
1	124	2022-01-01	This value	
2	125	2022-01-01		That value

Lab: Explore Azure Storage

In this lab, you will provision and use Azure Storage

1. Start the virtual machine for this lab
   or go to the exercise page at https://aka.ms/dp900-storage-lab
2. Follow the instructions to complete the exercise on Microsoft Learn
   Use the Azure subscription provided for this lab

1: Knowledge check

What are the elements of an Azure Table storage key?

- ❑ Table name and column name
- [✓ Partition key and row key]
- ❑ Row number

What should you do to an existing Azure Storage account in order to support a data lake for Azure Synapse Analytics?

- ❑ Add an Azure Files share
- ❑ Create Azure Storage tables for the data you want to analyze
- [✓ Upgrade the account to enable *hierarchical namespace* and create a blob container]

Which Azure Storage option should use to create cloud-based network file shares?

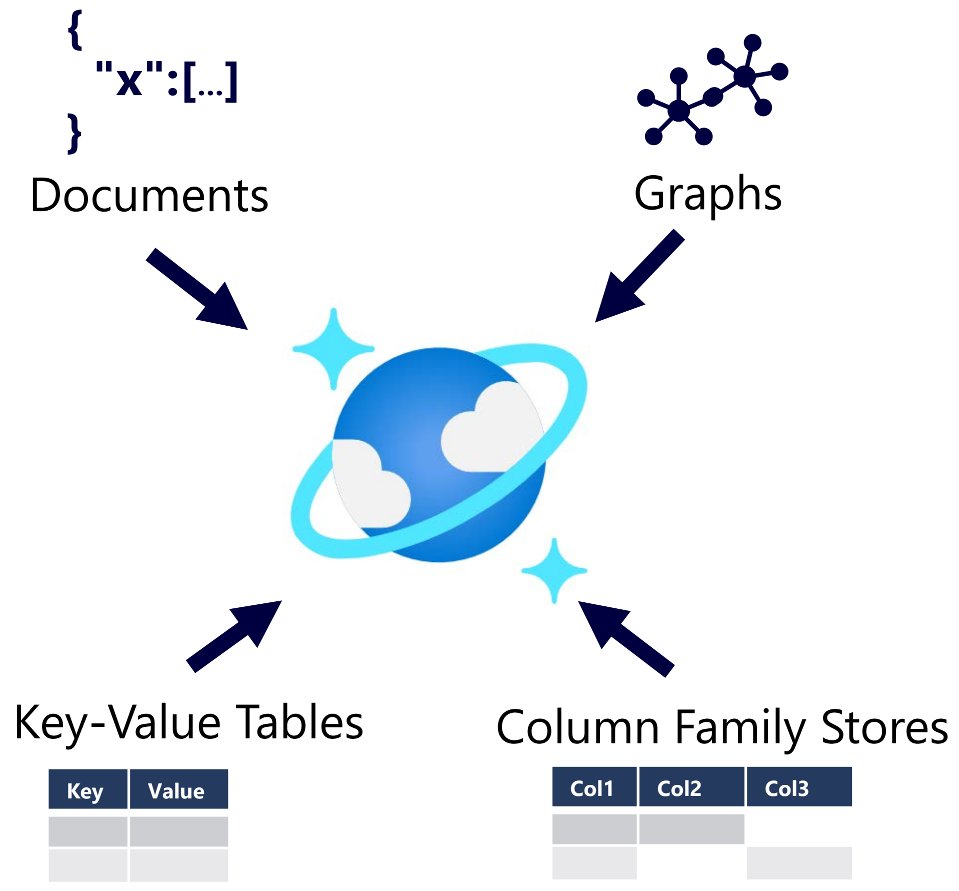
- ❑ Azure Blob Storage
- ❑ Azure Tables
- [✓ Azure Files]

2: Fundamentals of Azure Cosmos DB

What is Azure Cosmos DB?

A multi-model, global-scale *NoSQL* database management system

- Support for multiple storage APIs
- Real time access with fast read and write performance
- Enable *multi-region writes* to replicate data globally; enabling users in specified regions to work with a local replica



Azure Cosmos DB APIs

Azure Cosmos DB for NoSQL

- Native API for Cosmos DB

```
SELECT *
FROM customers c
WHERE c.id = "joe@litware.com"
```

```
{
   "id": "joe@litware.com",
   "name": "Joe Jones",
   "address": {
        "street": "1 Main St.",
        "city": "Seattle"
   }
}
```

Azure Cosmos DB for MongoDB

- Compatibility with MongoDB

```
db.products.find({ id: 123})
```

```
{
   "id": 123,
   "name": "Hammer",
   "price": 2.99}
}
```

Azure Cosmos DB for PostgreSQL

- Compatibility with PostgreSQL

id	name	dept	manager
1	Sue Smith	Hardware	Joe Jones
2	Ben Chan	Hardware	Sue Smith

Azure Cosmos DB for Table

- Key-value storage API
- Compatible with Azure Table Storage

PartitionKey	RowKey	Name
1	123	Joe Jones
1	124	Samir Nadoy

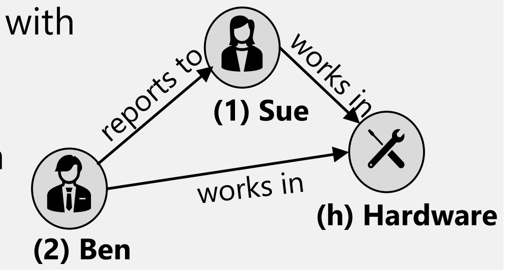
Azure Cosmos DB for Apache Cassandra

- Compatibility with Apache Cassandra

id	name	dept	manager
1	Sue Smith	Hardware	
2	Ben Chan	Hardware	Sue Smith

Azure Cosmos DB for Apache Gremlin

- Used to work with *graph* data
- *vertices* are connected via relationships (*edges*)



Lab: Explore Azure Cosmos DB

In this lab, you will provision and use Azure Cosmos DB

1. Start the virtual machine for this lab
   or go to the exercise page at https://aka.ms/dp900-cosmos-lab
2. Follow the instructions to complete the exercise on Microsoft Learn
   Use the Azure subscription provided for this lab

2: Knowledge check

Which Cosmos DB API should you use to store and query JSON documents in Azure Cosmos DB?

- ✓ Azure Cosmos DB for NoSQL
- ❑ Azure Cosmos DB for Apache Cassandra
- ❑ Azure Cosmos DB for Table

Which Azure Cosmos DB API should you use to work with data in which entities and their relationships to one another are represented in a graph using vertices and edges?

- ❑ Azure Cosmos DB for MongoDB
- ❑ Azure Cosmos DB for NoSQL
- ✓ Azure Cosmos DB for Apache Gremlin

How can you enable globally distributed users to work with their own local replica of a Cosmos DB database?

- ❑ Create an Azure Cosmos DB account in each region where you have users
- ❑ Use the Table API to copy data to Azure Table Storage in each region where you have users
- ✓ Enable multi-region writes and add the regions where you have users

Microsoft Azure