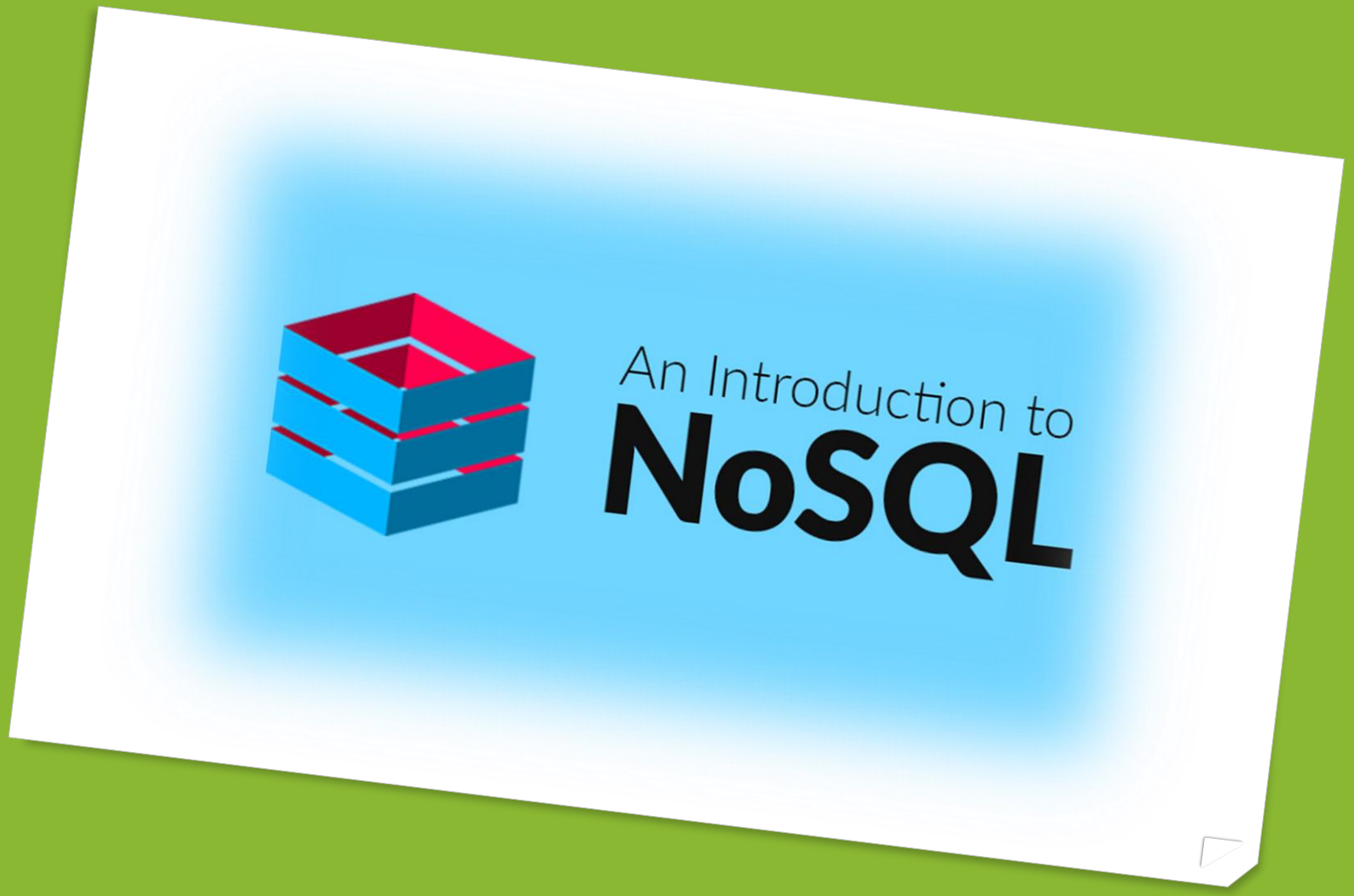
Azure NoSQL Offerings

Eshant Garg
Azure Data Engineer, Architect, Advisor
eshant.garg@gmail.com

Why NoSQL DB?

What traditional databases were lacking?

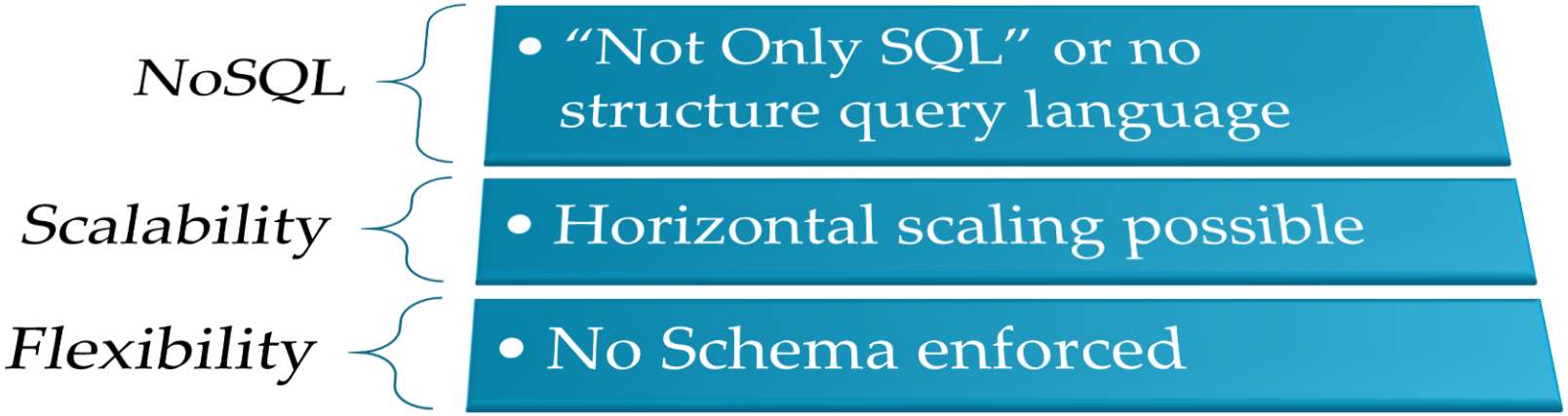


RDBMS were lacking

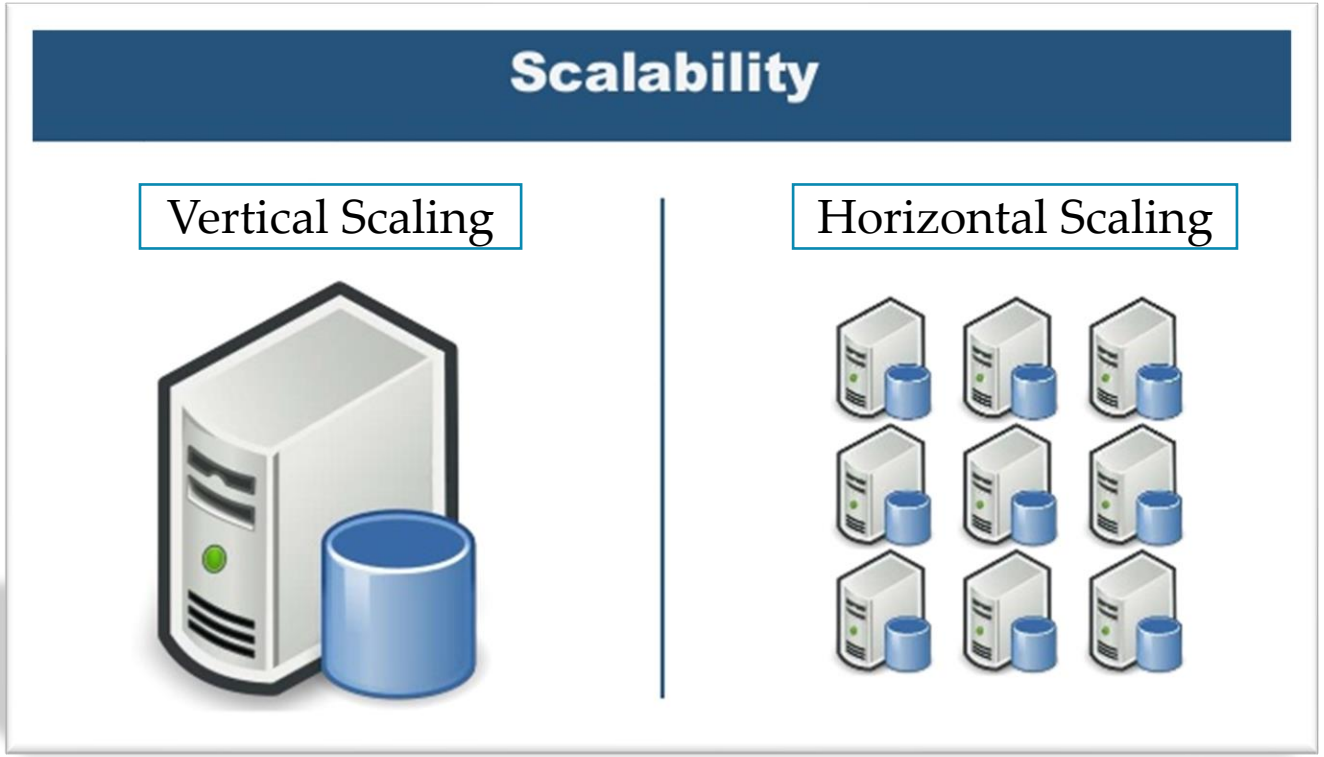
Scalability

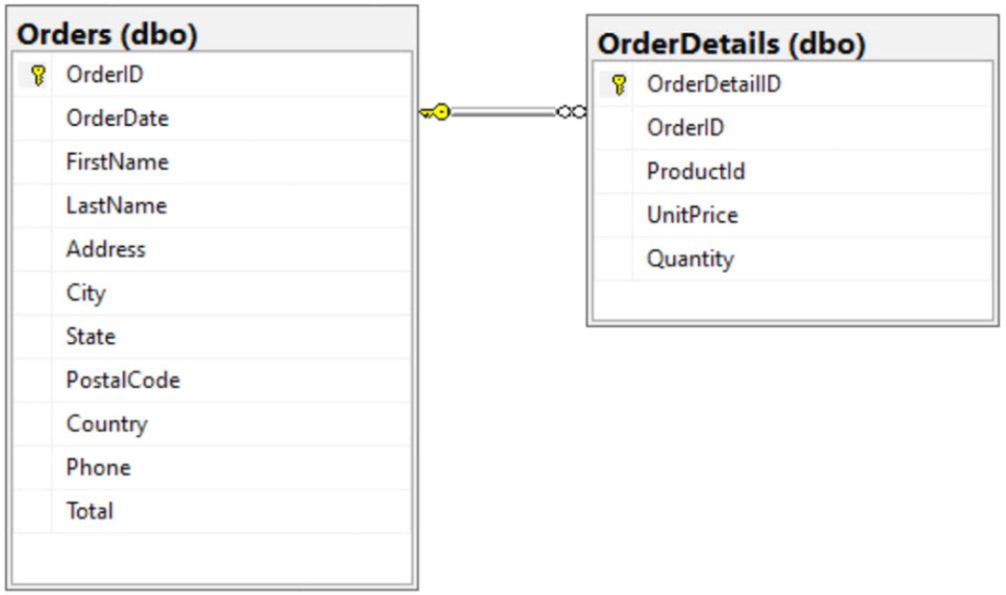
Flexibility

What is NoSQL



- **Vertical scaling**
  - Add more CPU, RAM, HDD in same system
- **Horizontal Scaling**
  - Add more commodity machines in system

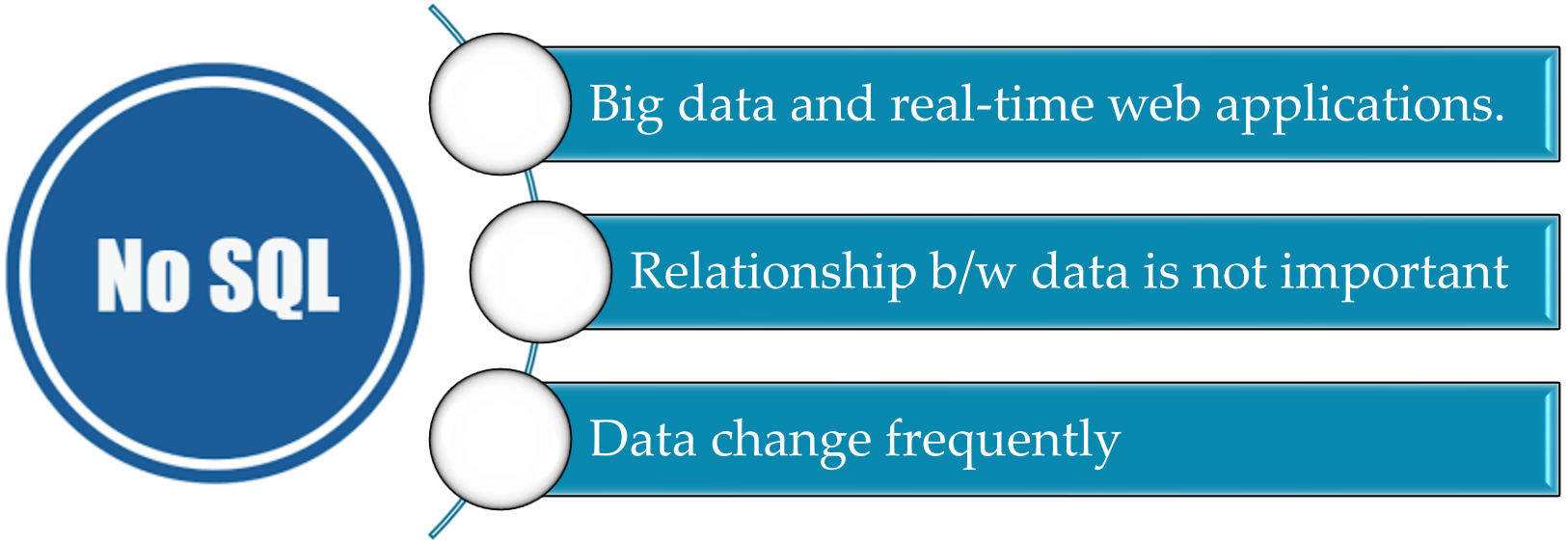





```
{
    "OrderId": 1,
    "OrderDate": 1574161910220,
    "FirstName": "John",
    "LastName": "Smith",
    "Address": "10 Street",
    "City": "City",
    "State": "VA",
    "OrderDetails": [
        {
            "UnitPrice": 7.99,
            "OrderDetailId": 2,
            "Quantity": 1,
            "ProductId": 259694,
            "OrderId": 1
        },
        {
            "UnitPrice": 7.99,
            "OrderDetailId": 3,
            "Quantity": 1,
            "ProductId": 295693,
            "OrderId": 1
        }
    ],
    "id": "795c50dc-1a83-11ea-bf07-00163ee85f66",
    "_rid": "VdgtAK23OMANAAAAAAAAAA==",
    "_self": "dbs/VdgtAA==/colls/VdgtAK23OMA=/docs/VdgtAK23OMANAAAAAAAAAA==/",
    "_etag": "\"370017e1-0000-1100-0000-5df770f20000\"",
    "_attachments": "attachments/",
    "_ts": 1576497394
}
```

```
{
  "orderid": 12212,
  "orderdate": "12/4/2020",
  "customer":
      { "name": "Bob Smith", "email": "bobsmith@email.bob" },
  "status": "in process",
  "paymentmethod": "invoice",
  "products": [
      { "name": "Product 1", "quantity": 1 },
      { "name": "Product 2", "quantity": 1, status: 3 }
  ]
}
```

NoSQL Use Cases



NoSQL Limitations

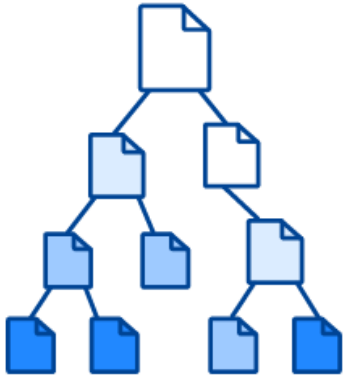
- Schema-less data means inconsistent data
- Denormalized data means redundant data
- Redundant data means inaccuracies and conflicts
- Does not support many good features of Relational DB
  - SPs, Functions, Views, Row level security, Locks, etc.

SQL vs NoSQL

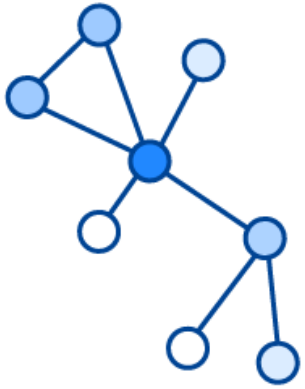
SQL	NoSQL
• Relational database • Fixed schema • Designed for complex queries • SQL, MySql, Oracle, Postgres • Vertical scaling • Row Oriented • Tables • Limited for big data	• Non-relational or distributed • Dynamic • Not for complex queries • MongoDB, Redis, Hbase • Horizontal scaling • Multi-model oriented • Collections • Great for big data

4 Types of NoSQL Databases

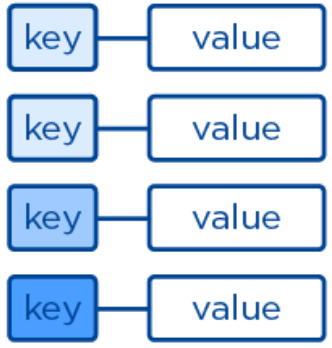
Document



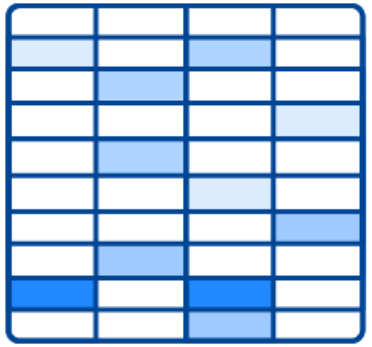
Graph



Key-Value

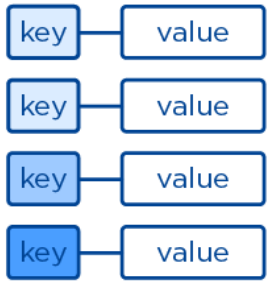


Wide-column



Key-value store

Key-Value

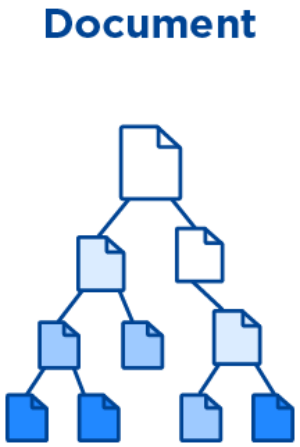


Phone Directory

Key	Value
Bob	(123) 456-7890
Jane	(234) 567-8901
Tara	(345) 678-9012
Tiara	(456) 789-0123

- Uses a simple key/value to store data
- Quick to query due to its simplicity
- Value can be JSON, BLOB, String etc.
- **Use Cases:**
  - User profiles and session info on a website, blog comments, telecom directories, IP forwarding tables, shopping cart contents on e-commerce sites, and more.
- **Examples**
  - Cosmos DB Table API, Redis, Table Storage, Oracle NoSQL Database, Voldemorte, Aerospike, Oracle Berkeley DB

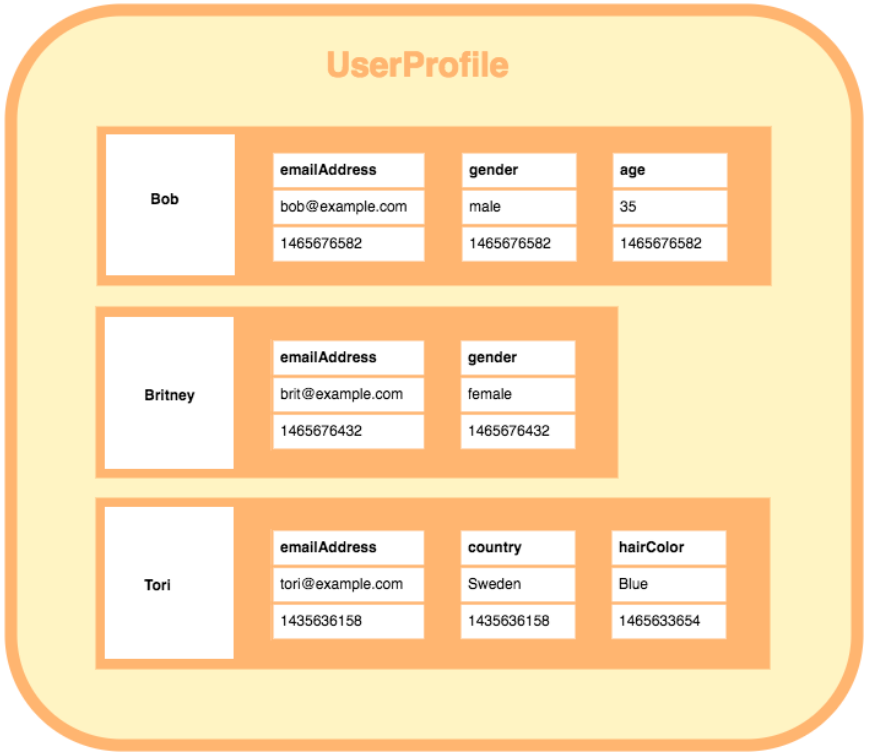
Document store



- Document-oriented model to store data
- Similar to key/value store, difference is that, the value in a document store database consists of semi-structured data.
- Each record and its associated data within a single document.
- Document stores are usually XML, JSON, BSON, YAML, etc.
- **Use Cases:**
  - Content management systems, blogging platforms, and other web applications, blog comments, chat sessions, tweets, ratings, etc.
- **Examples**
  - Cosmos DB, MongoDB, DocumentDB, CouchDB, MarkLogic, OrientDB

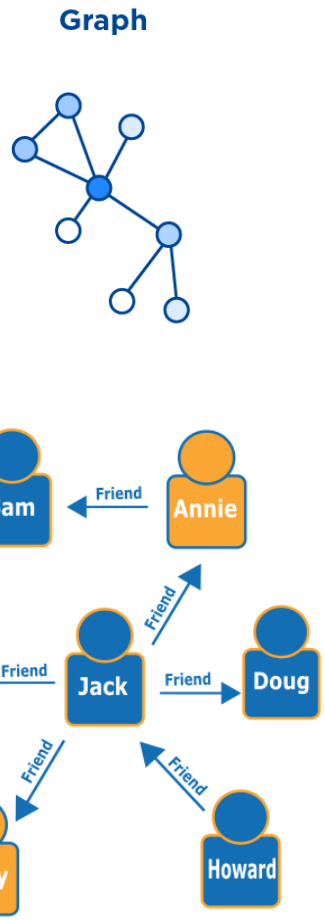
```
{
  "orderid": 12212,
  "orderdate": "12/4/2020",
  "customer":
      { "name": "Bob Smith", "email": "bobsmith@email.bob" },
  "status": "in process",
  "paymentmethod": "invoice",
  "products": [
      { "name": "Product 1", "quantity": 1 },
      { "name": "Product 2", "quantity": 1, status: 3 }
  ]
}
```

Column store



- Stores data using a column oriented model
- Columns in each row are contained within that row
- Each row can have different columns to the other rows.
- Extremely quick to load and query
- **Use Cases:**
  - Sensor Logs [Internet of Things (IOT)], User preferences, Geographic information, Reporting systems, Time Series Data, Logging and other write heavy applications
- **Examples**
  - Cosmos DB, Bigtable, Cassandra, Hbase, Vertica, Druid, Accumulo, Hypertable

Graph store



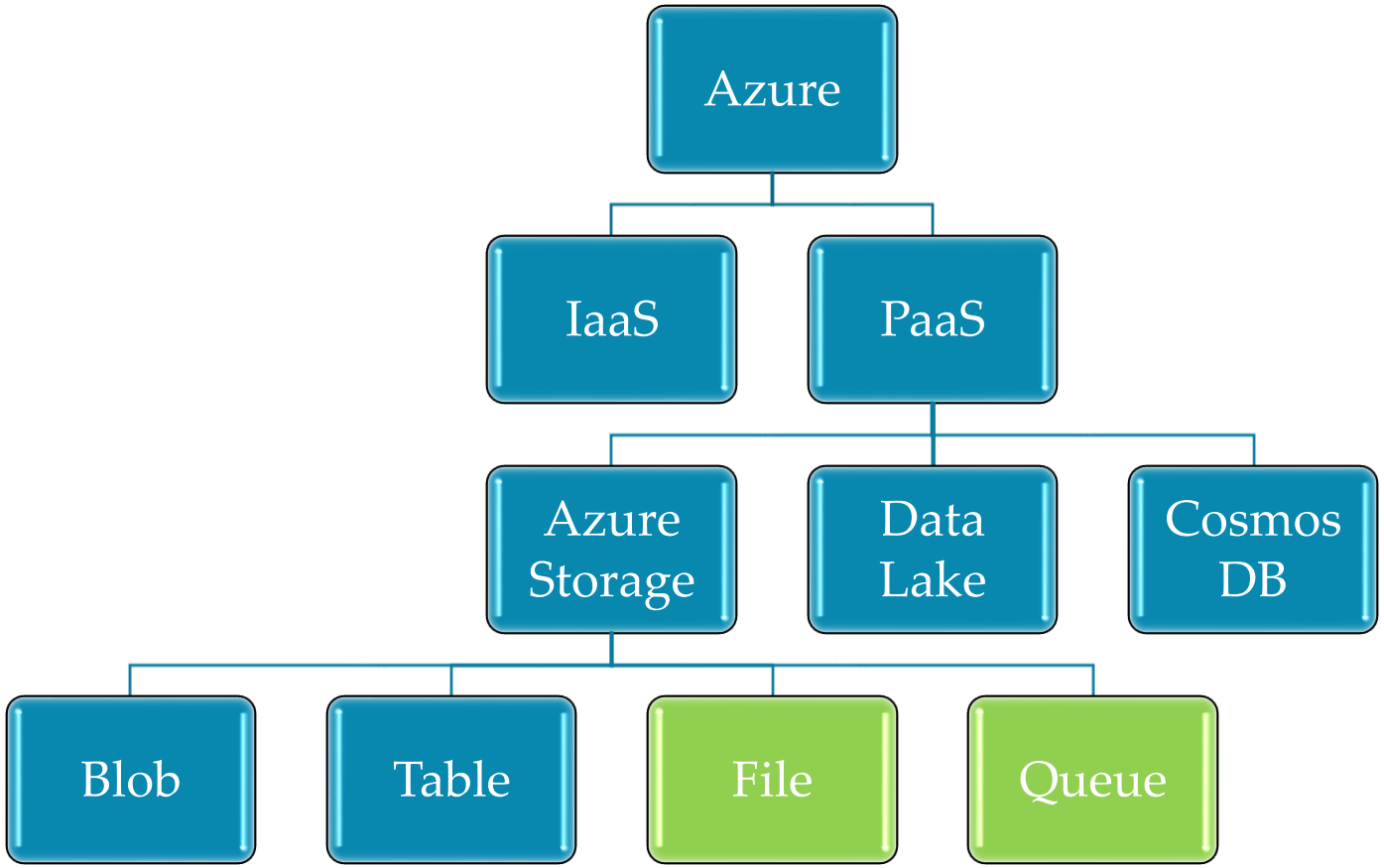
- Focuses on how data relates to other data points.
- A **node** is a specific entity or piece of information
- **Edge** simply specifies the relationship between two nodes.
- **Use Cases:**
  - Social networks, realtime product recommendations, network diagrams, fraud detection, access management, and more.
- **Examples**
  - Cosmos DB Gremlin API, Neo4j, Blazegraph, and OrientDB.

Multi-model

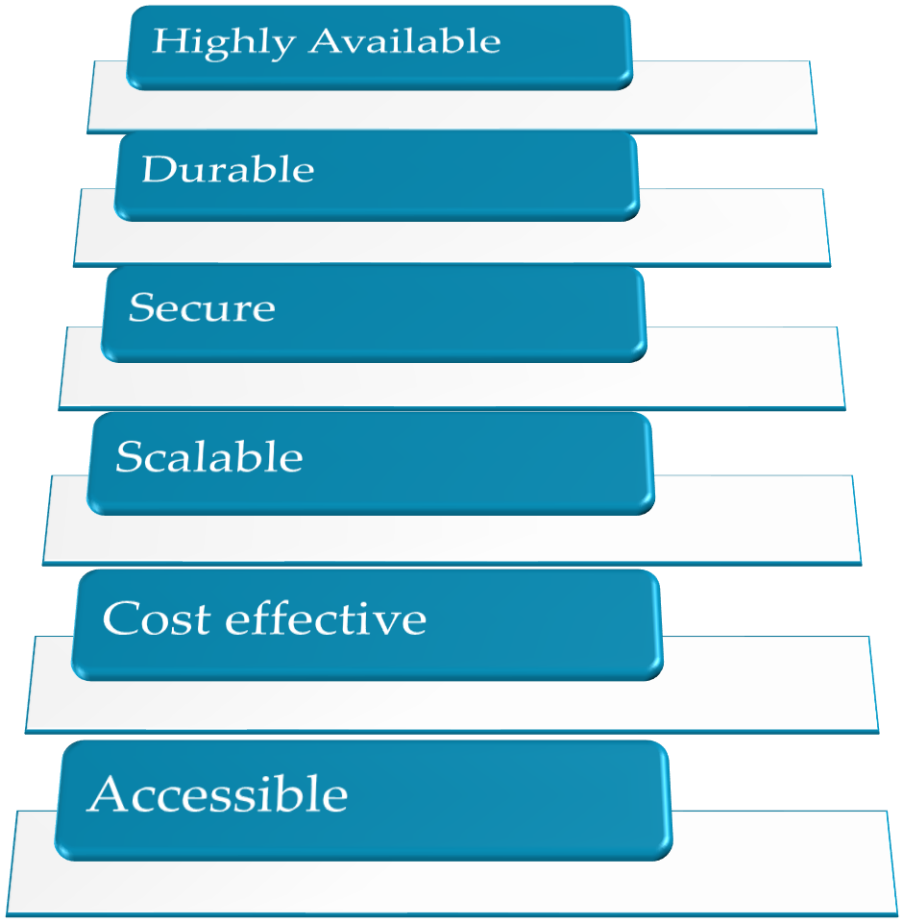
- Include features/characteristics of more than one data model.

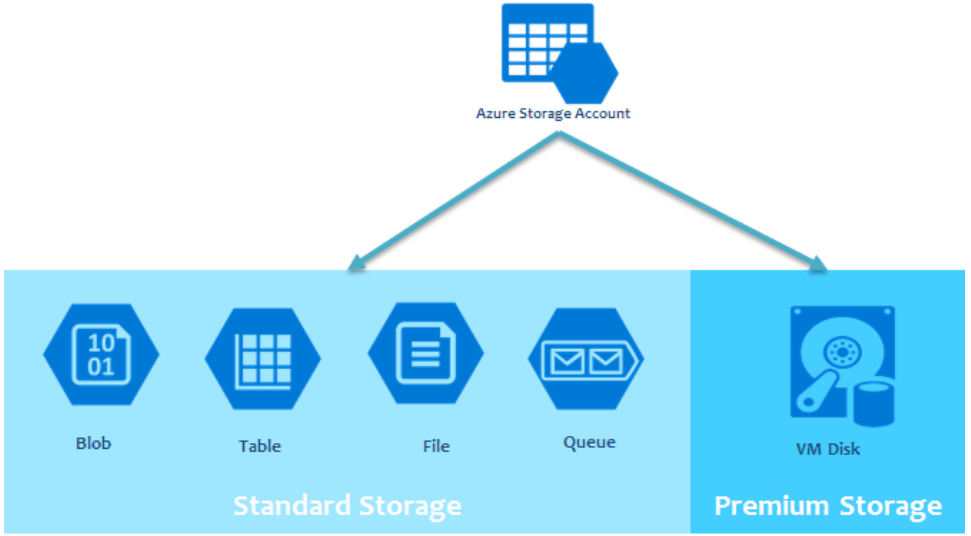
- **Example:**
  - **OrientDB:** OrientDB combines a graph model with a document model.
  - **ArangoDB:** Uses key/value, document, and graph models.
  - **Virtuoso:** Combines relational, graph, and document models.

NoSQL Offerings by Microsoft Azure

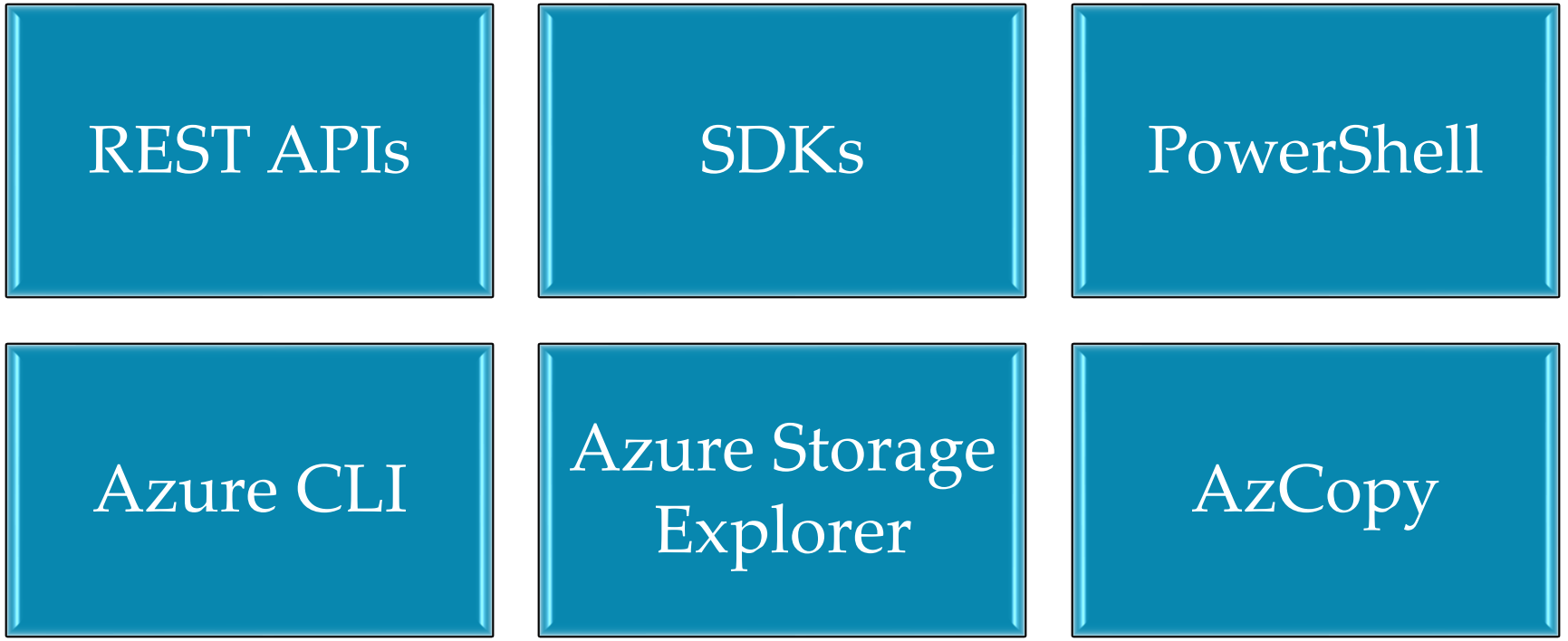


Microsoft Azure Storage

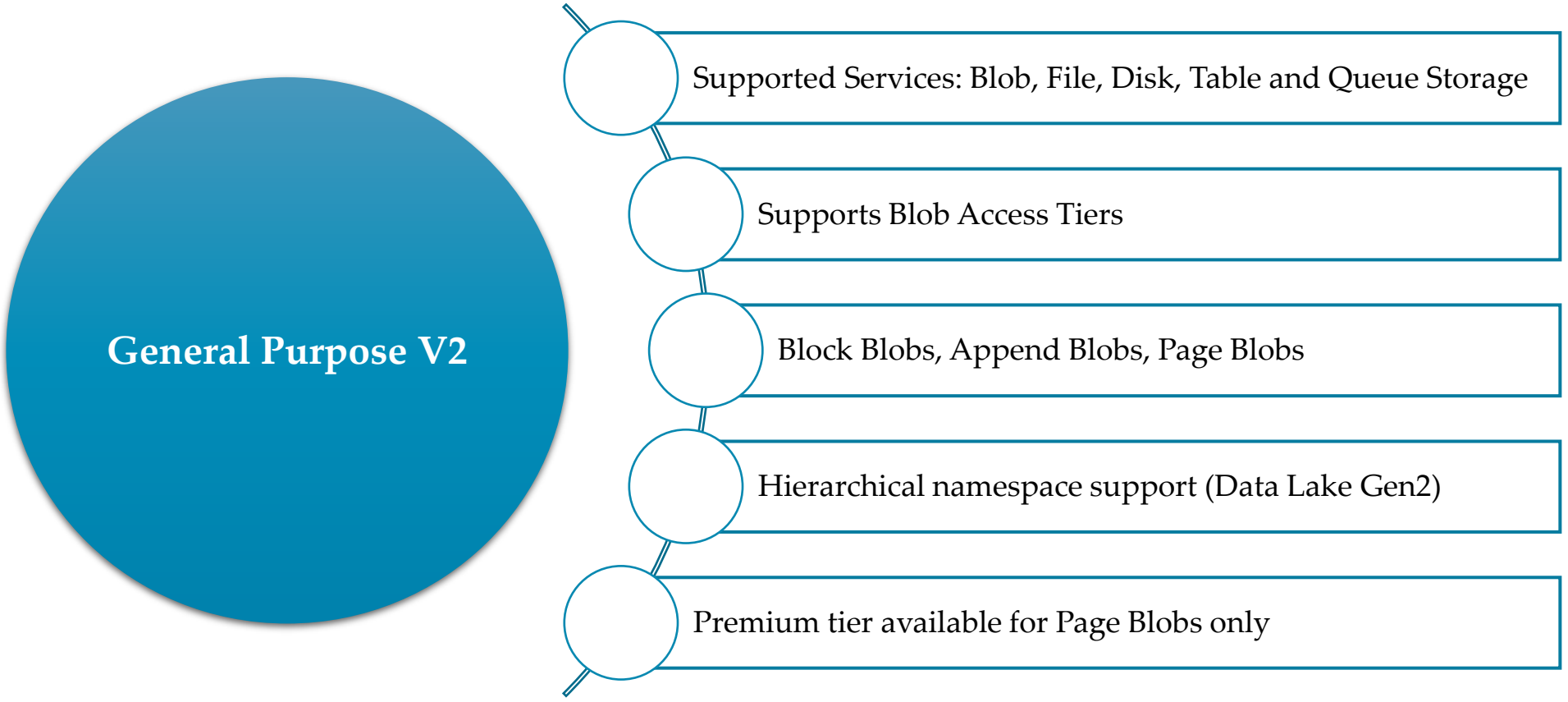




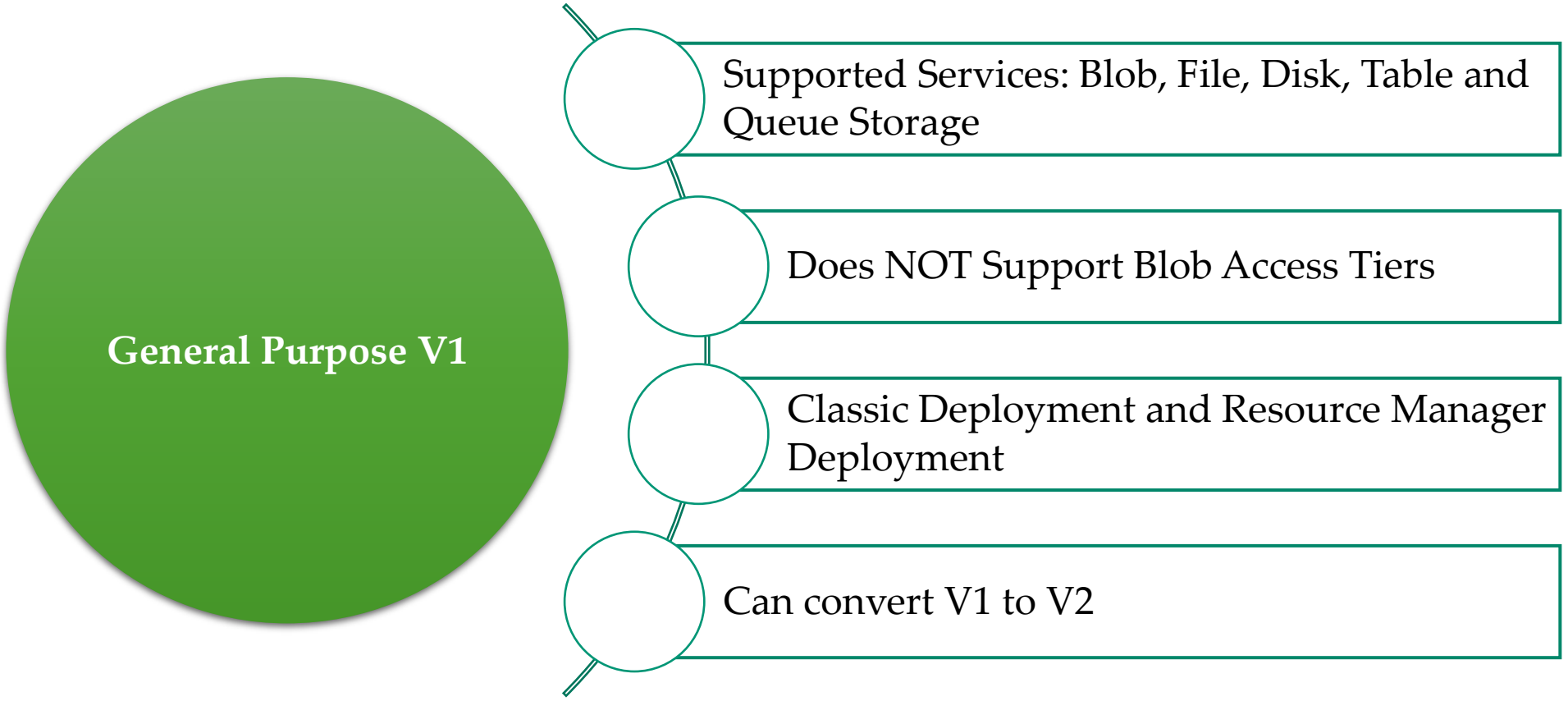
Programmatic Access to Storage Accounts



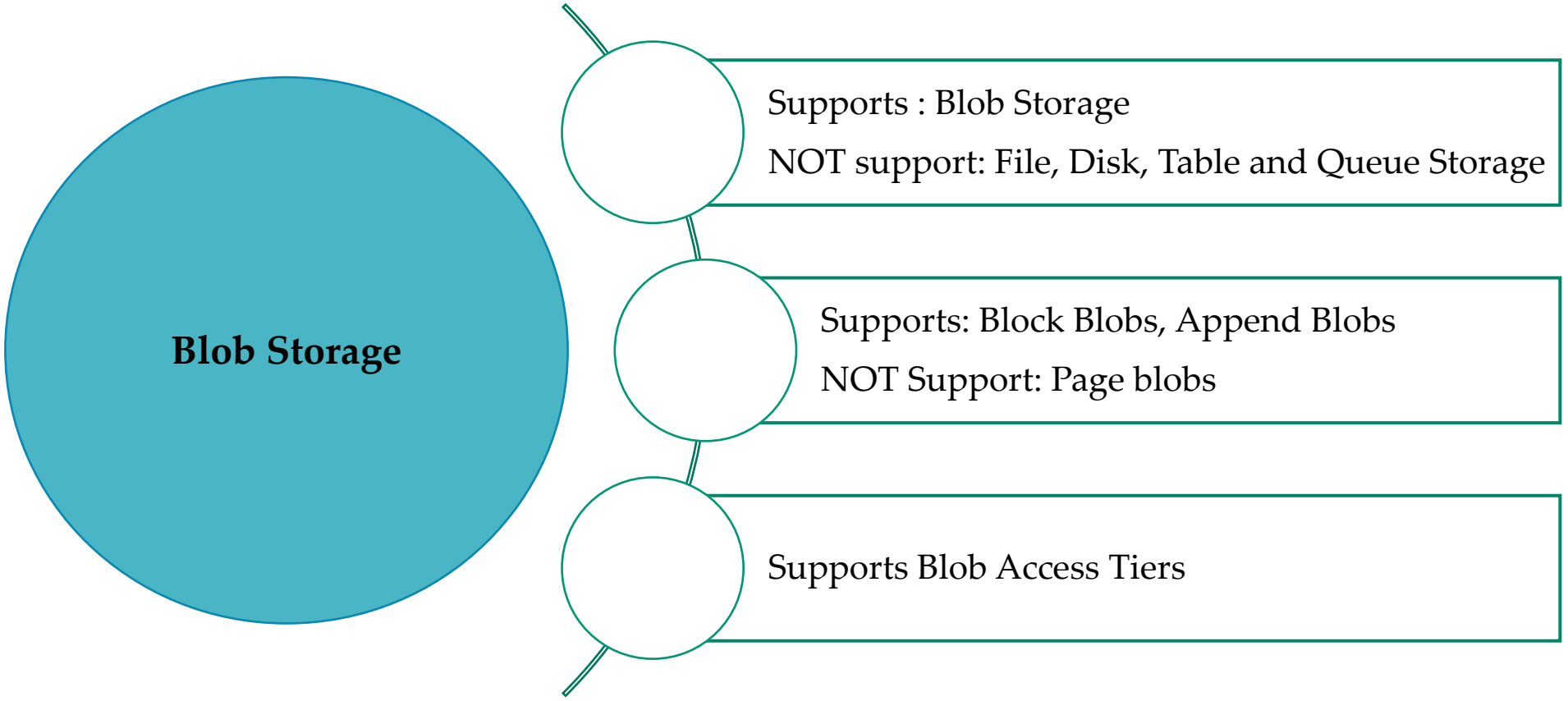
Azure Storage Account Type



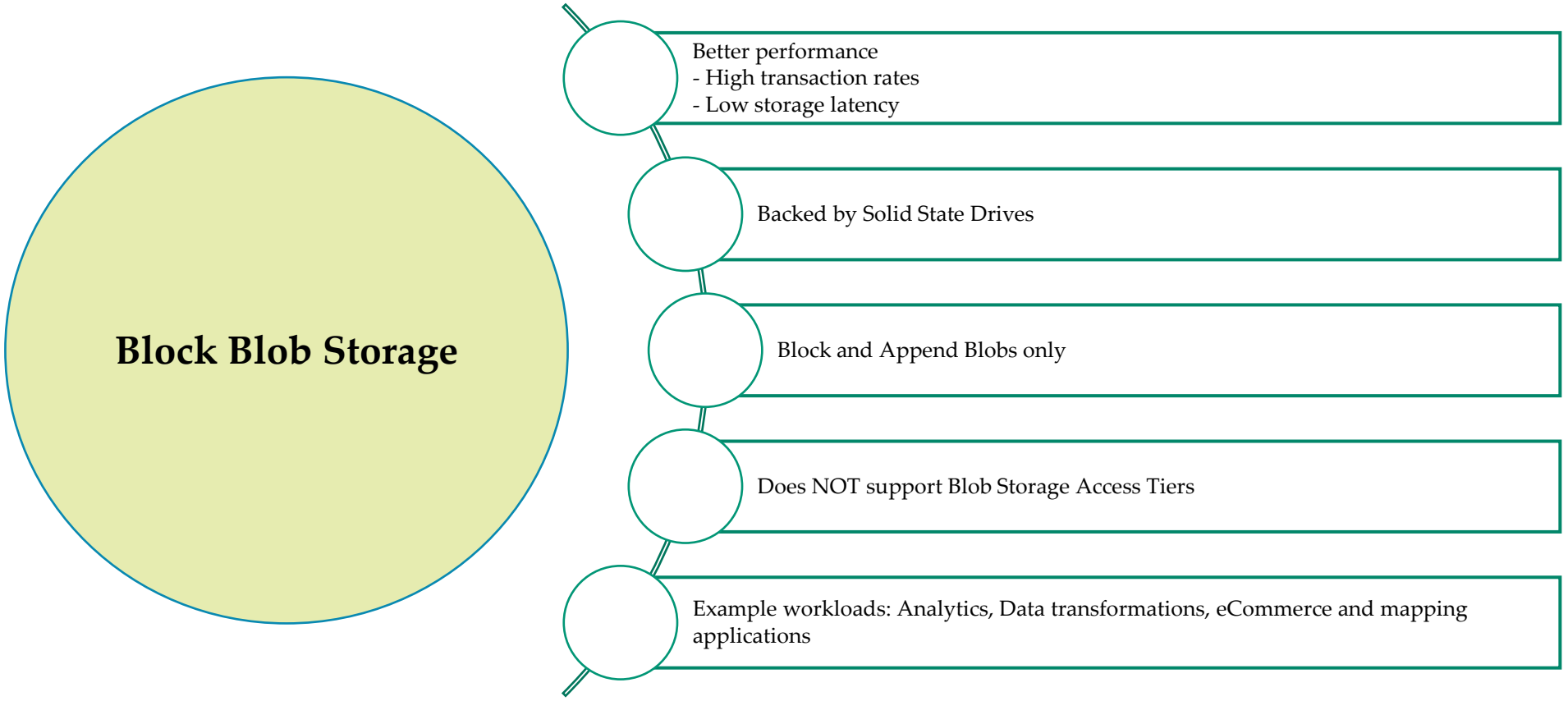
Azure Storage Account Type



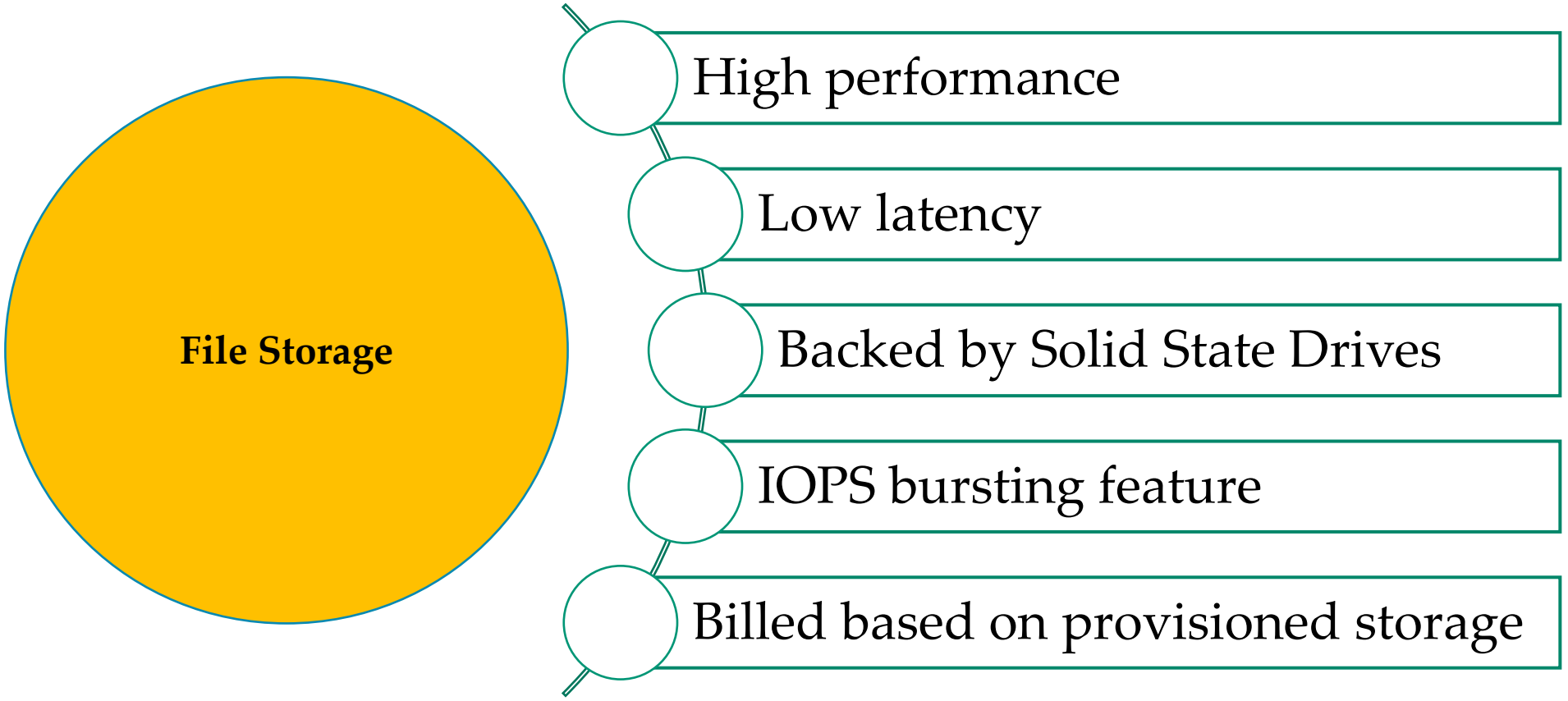
Azure Storage Account Type



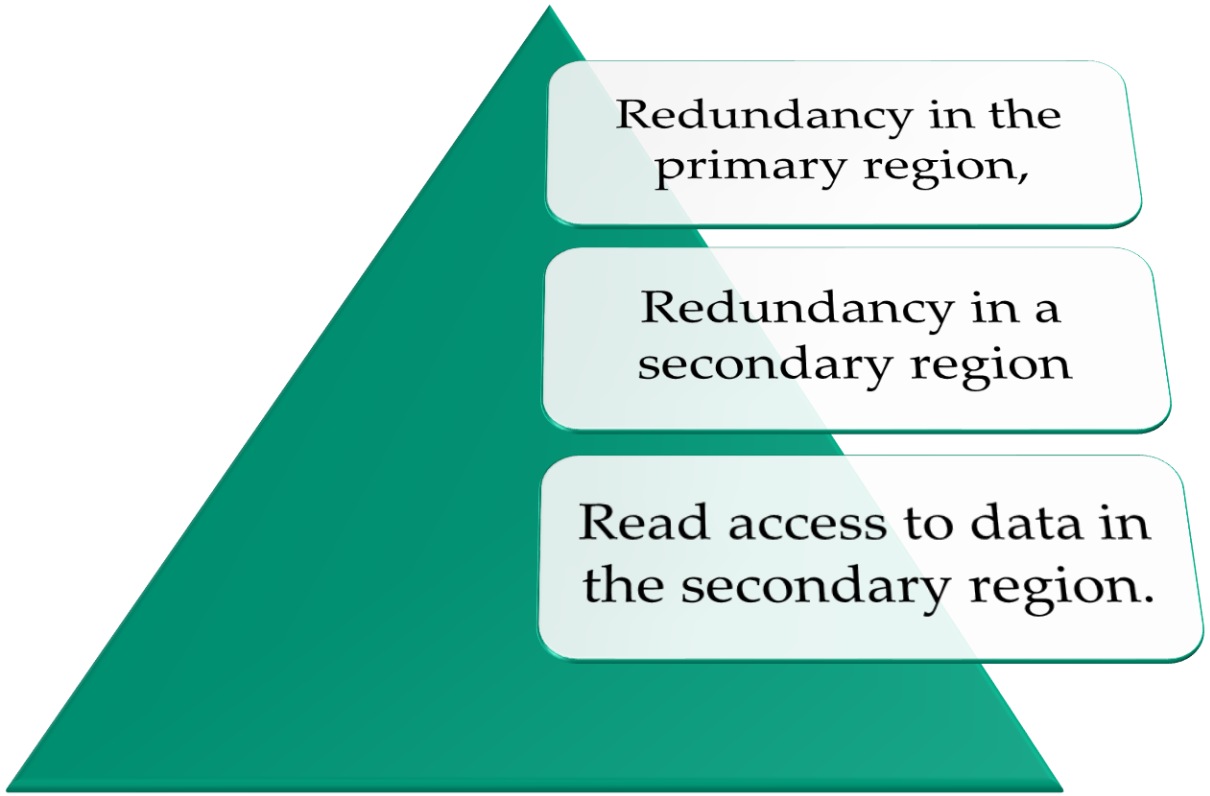
Azure Storage Account Type



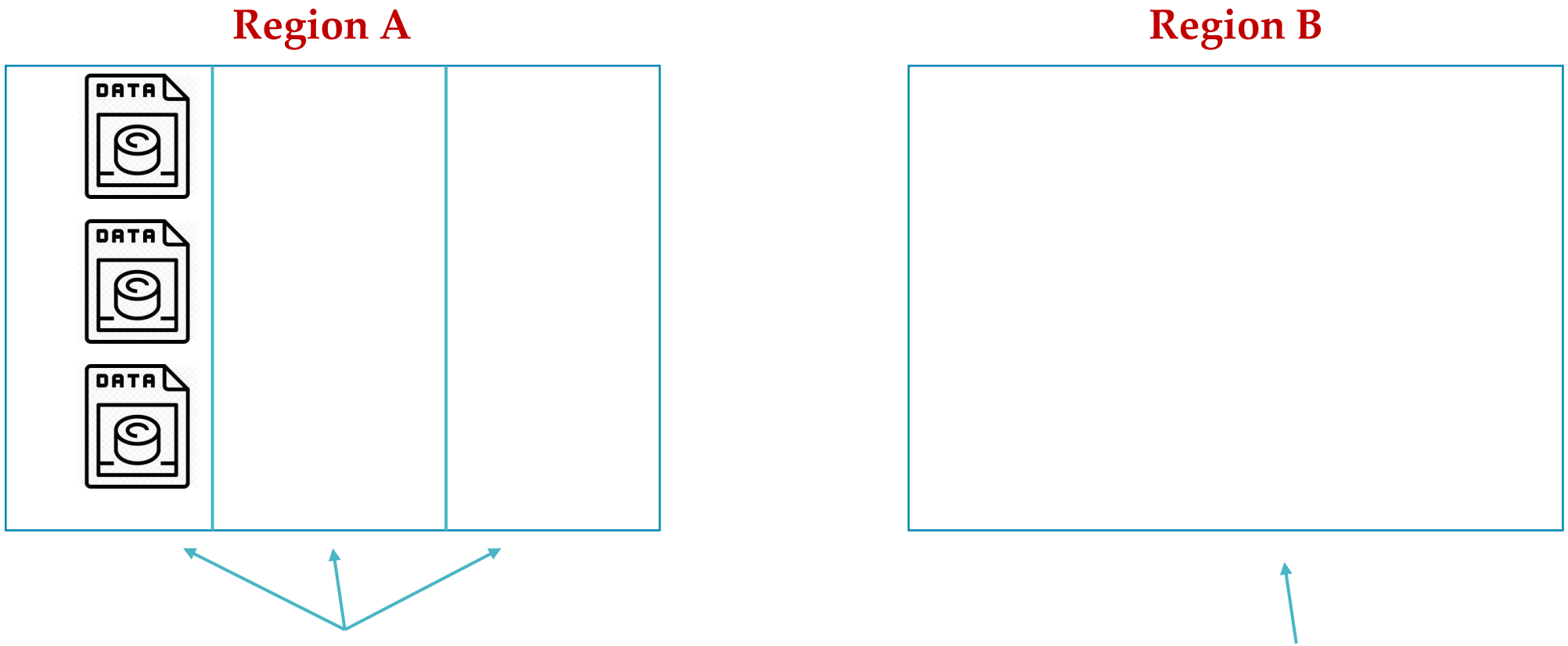
Azure Storage Account Type



Three categories of replication options



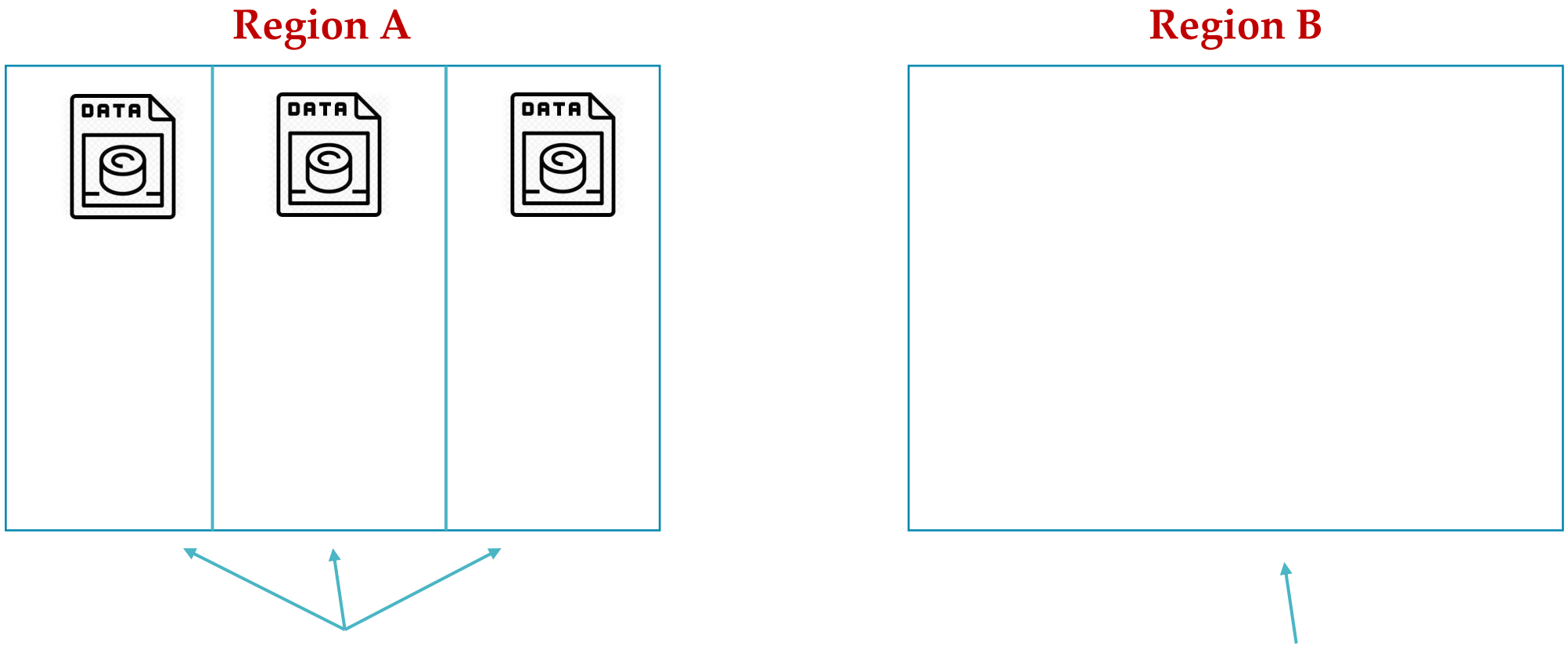
Locally Redundant Storage (LRS)



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

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

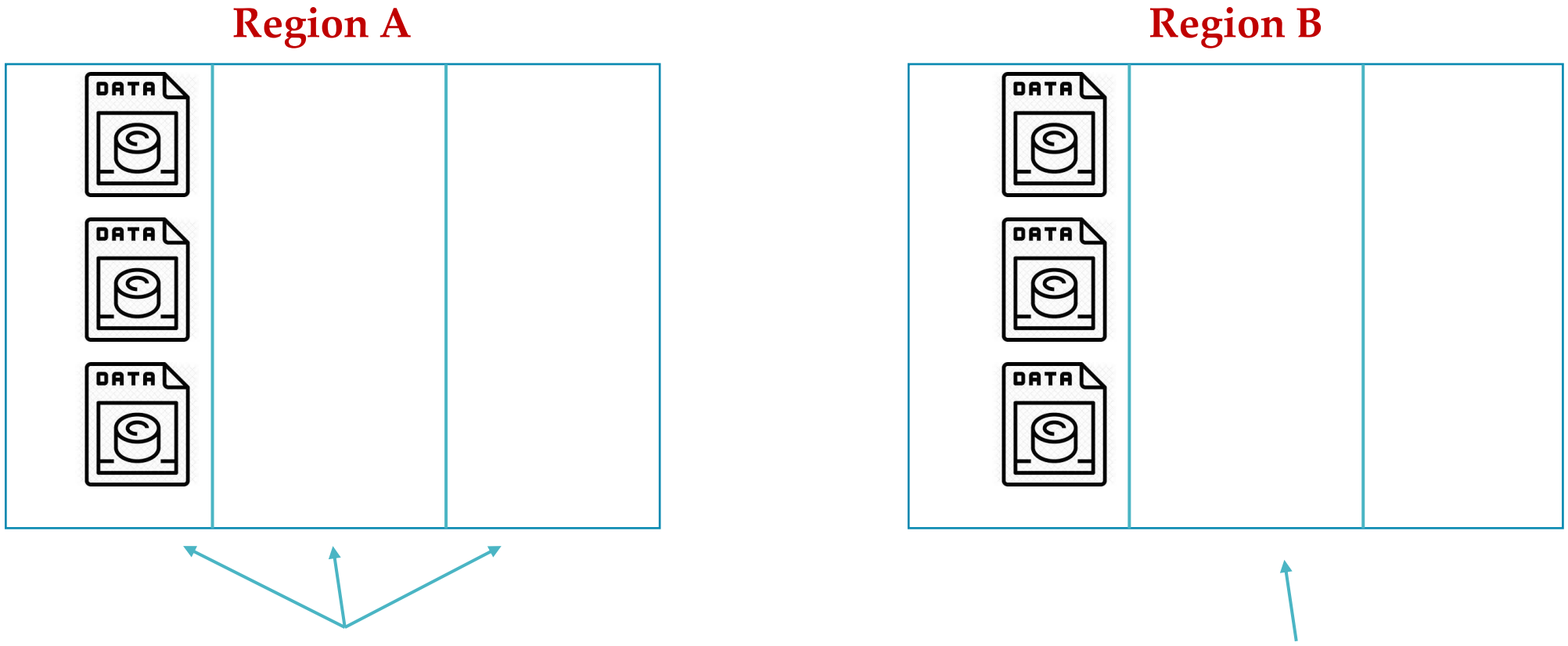
Zone Redundant Storage (ZRS)



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

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

Geo Redundant Storage (GRS)



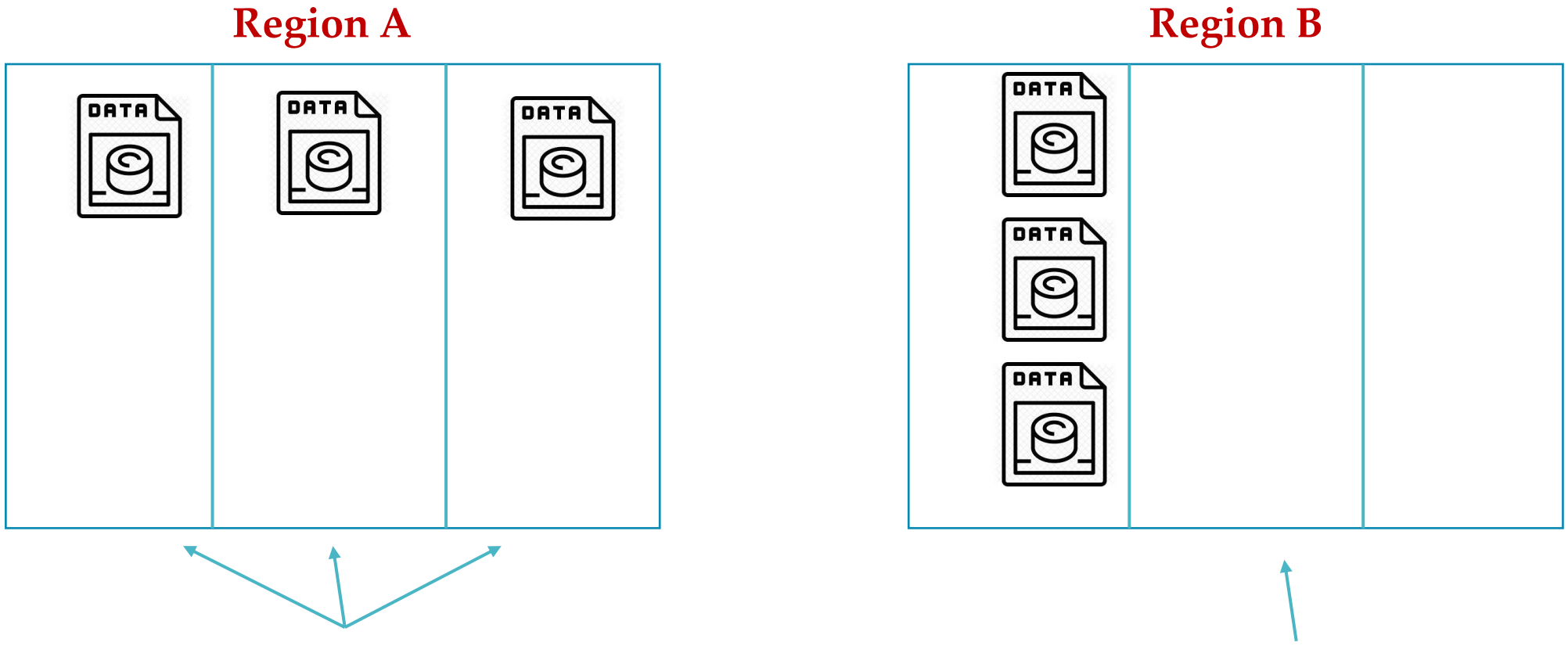
Storage Clusters

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

Region B

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

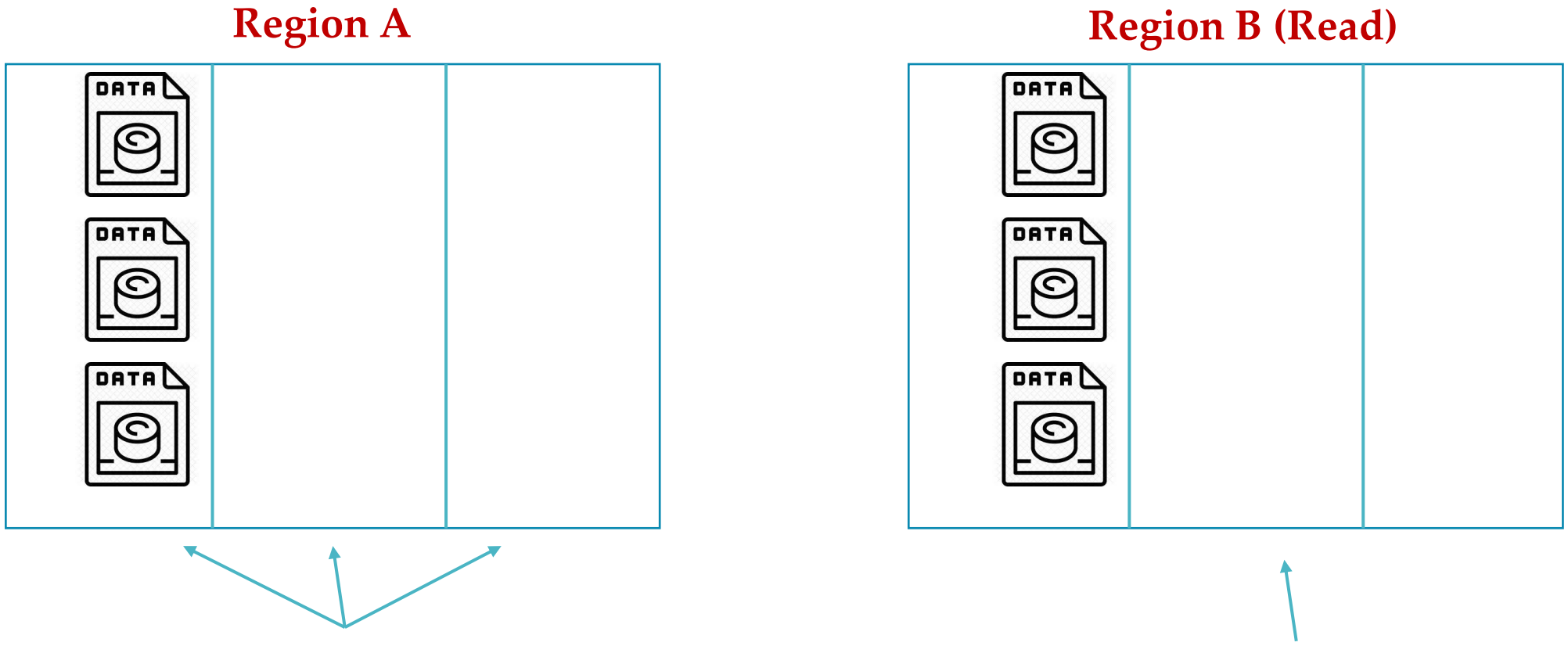
Geo Zone Redundant Storage (GZRS)



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

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

Read access geo Redundant Storage (RA-GRS)



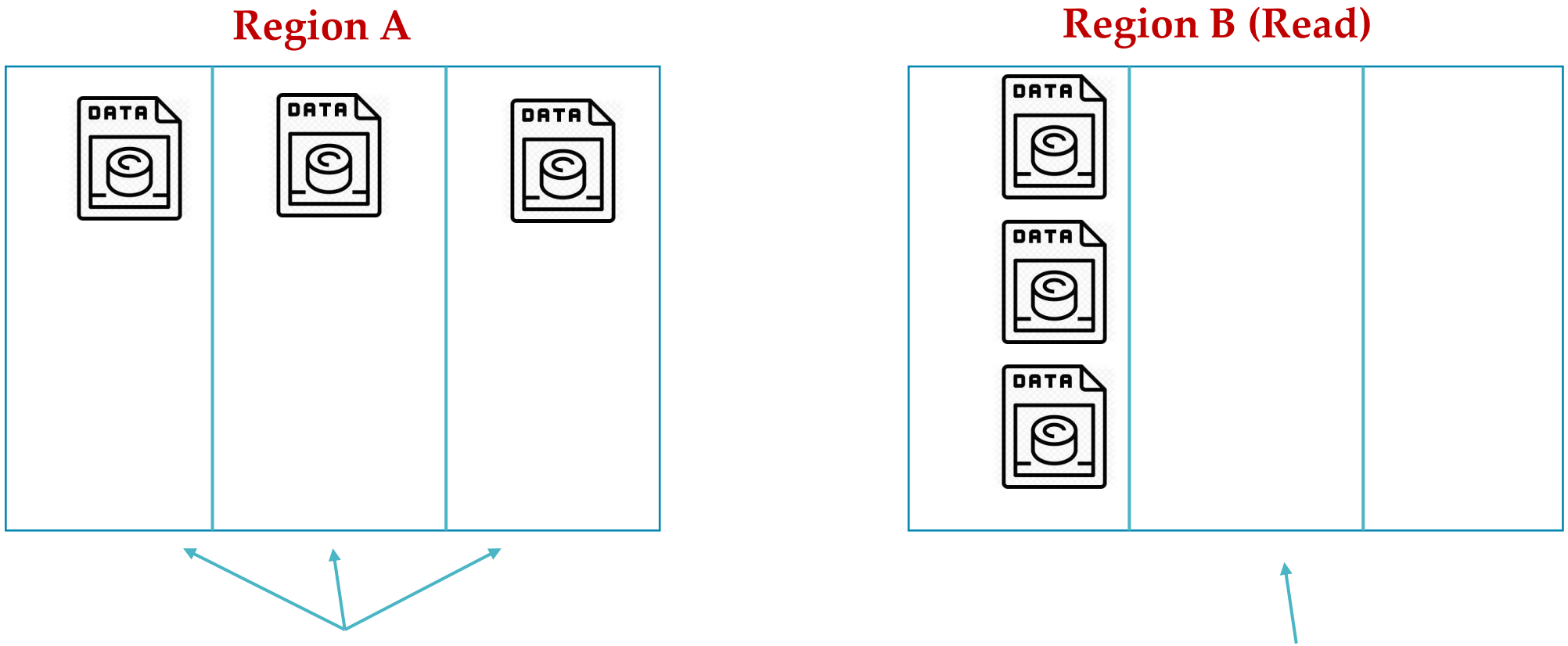
Storage Clusters

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

Region B

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

Read access Geo Zone Redundant Storage (RA-GZRS)



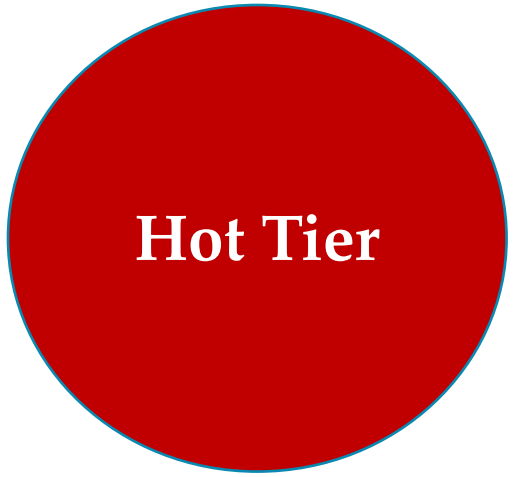
Storage Clusters

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

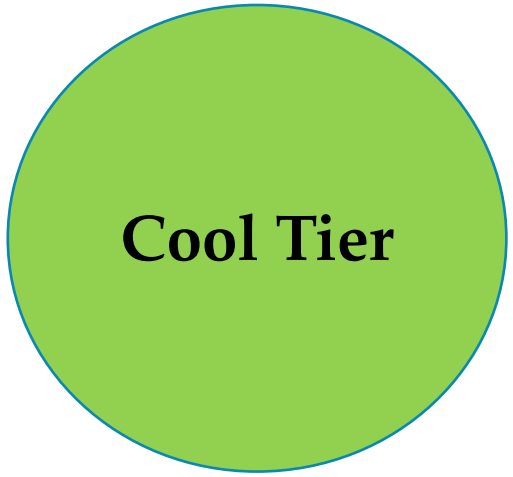
Region B

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

Blob Access Tiers



Highest storage cost
Lowest data access cost



Lowest storage cost
Higher data access cost

Lowest storage cost
Highest data retrieval cost
Data is offline

Azure Blob Storage

- Designed for images and unstructured Data
  - Store Documents and access in browser
  - Database backup
  - Store audio and video files and stream them
  - Store data for analysis
  - Log files
- Scalability
- Cheapest way to store data in azure
- Simple design and easy to use
- HDFS and blob storage REST APIs

Blob Types

- **Block Blog**
  - Composed for Blocks
- **Append Blob**
  - Can only append blocks
  - Ideal for logs
- **Page Blob**
  - VM disks and databases
  - Frequent random read/write applications

Use cases

- Only basic storage is needed
- Data is unstructured
- Data that is older or not used as much
- Money is an issue

Advantages of Blob storage

- Extremely cheap
- Simple to setup
- No configuration
- Doesn't require powerful computing to manage

Limitations of Blob storage

- No Indexes
- No Search Tools
- Not optimized for performance
- You are responsible for replication and synchronization
- Requires external compute to process