AZURE COSMOS DB

Develop Cloud-Native Intelligent Apps at Scale

NoSQL

Overview

Optimized for data in the modern world

Useful for data that is:

- Geographically distributed
- Big data
- Updated frequently

NoSQL databases read, query, and write data at speeds that relational database cannot attain

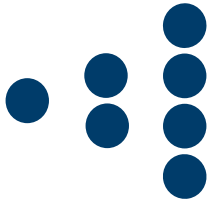
NoSQL

Overview

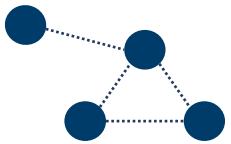
- Focuses on performance over consistency
- Allows data to have structure without enforcing Schema
- Data is replicated across many nodes asynchronously
- Types of NoSQL databases:

Key-value

Column-family



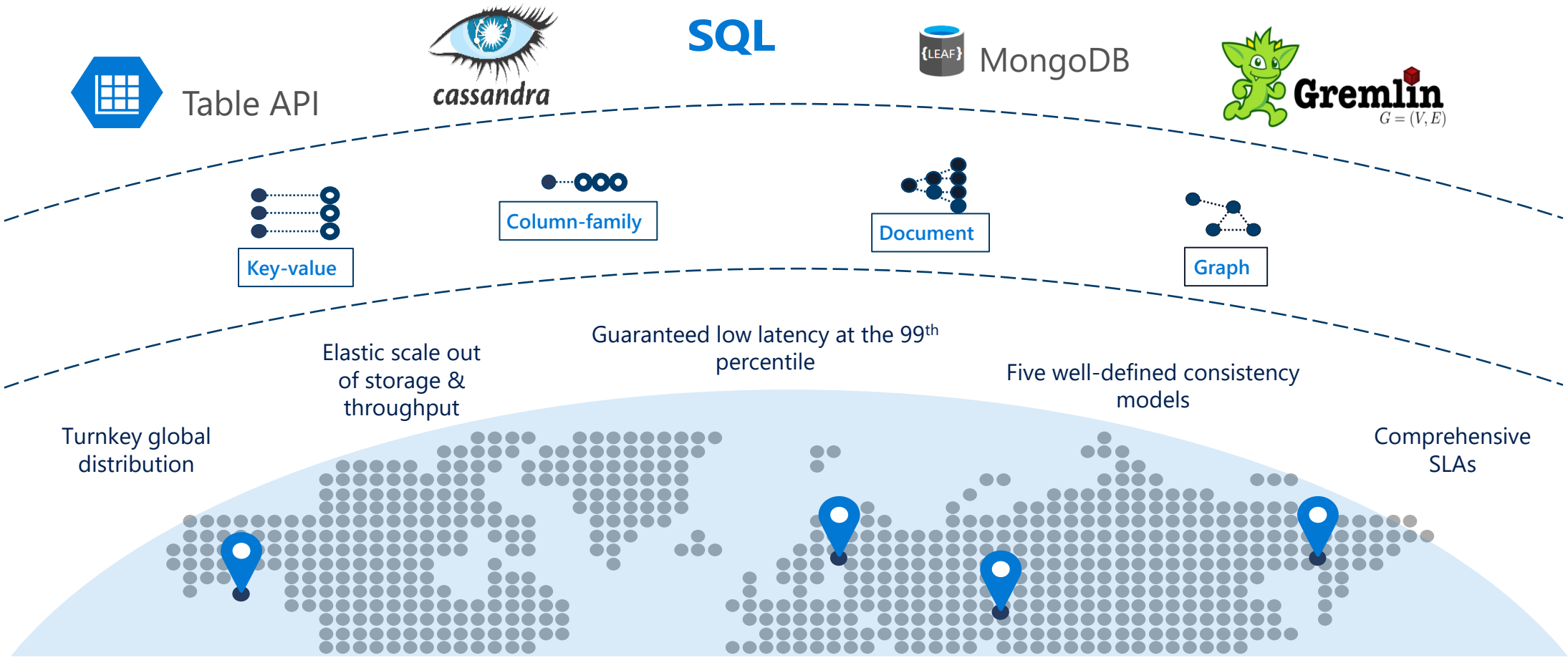
Document



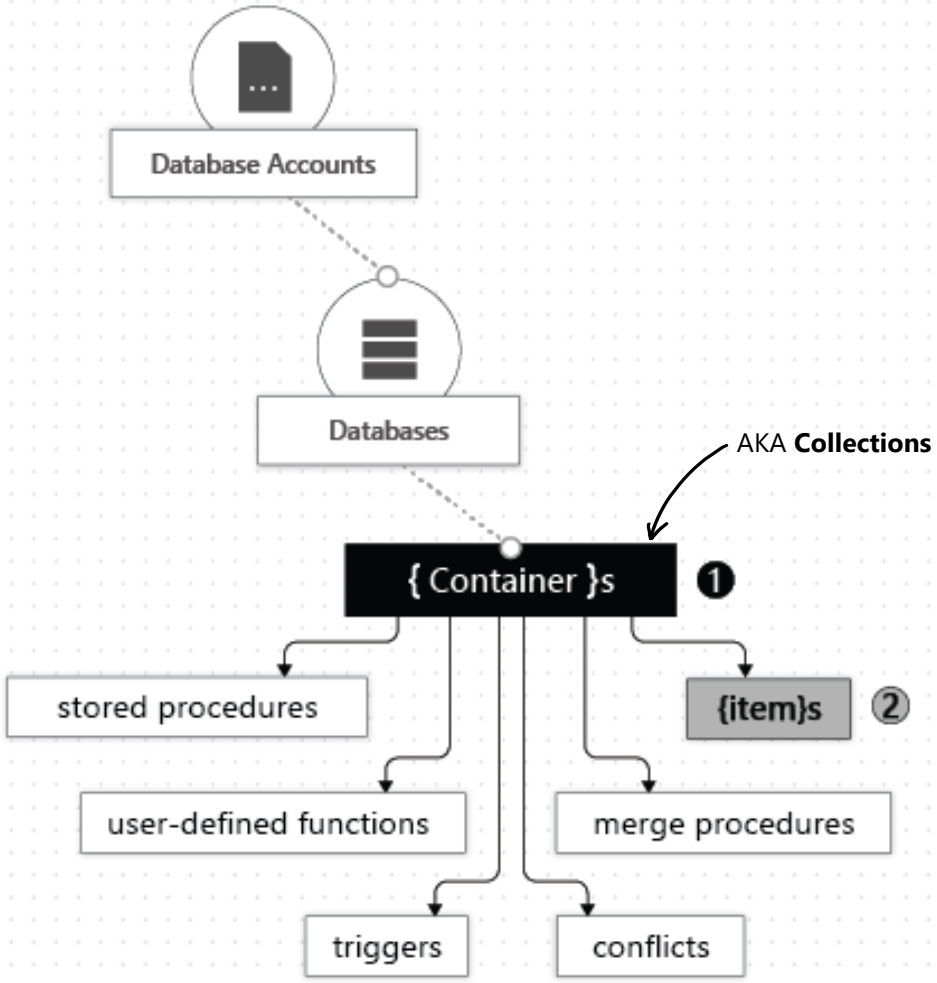
Graph

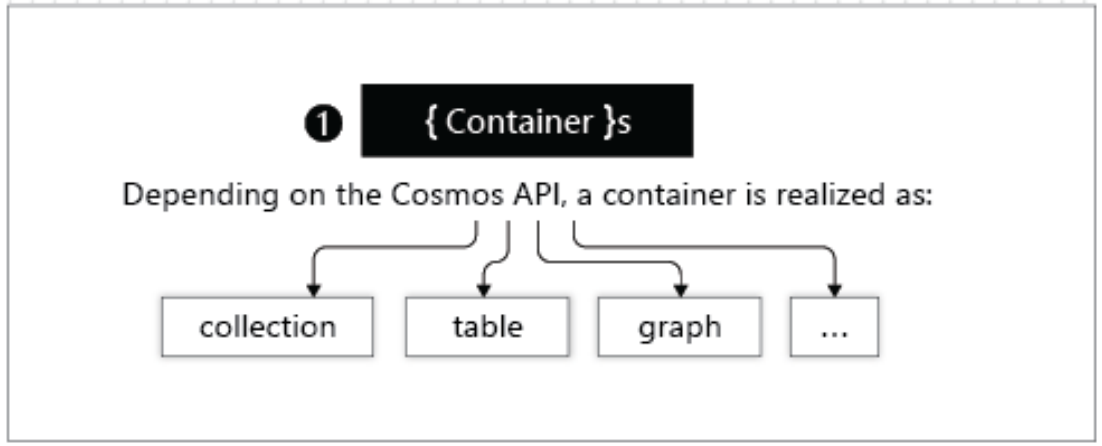
Azure Cosmos DB

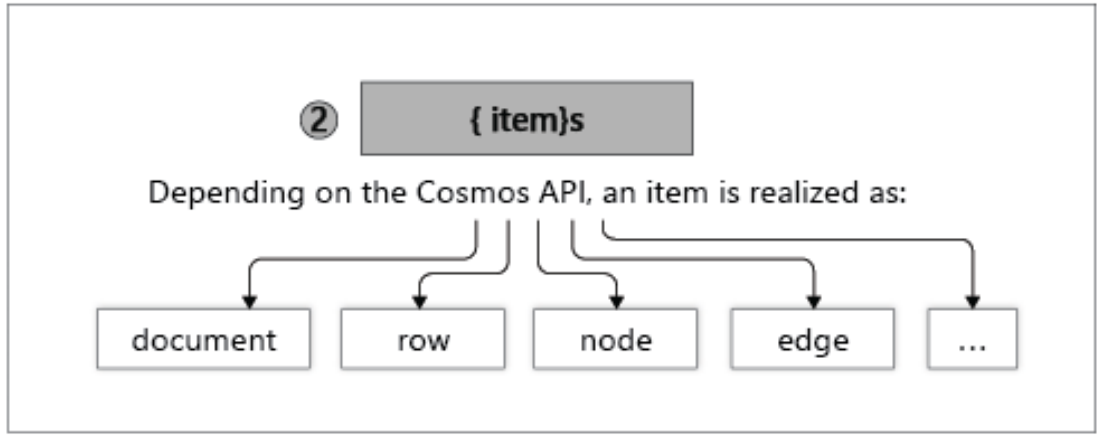
A globally distributed, massively scalable, multi-model database service



Azure Cosmos DB Resource Model



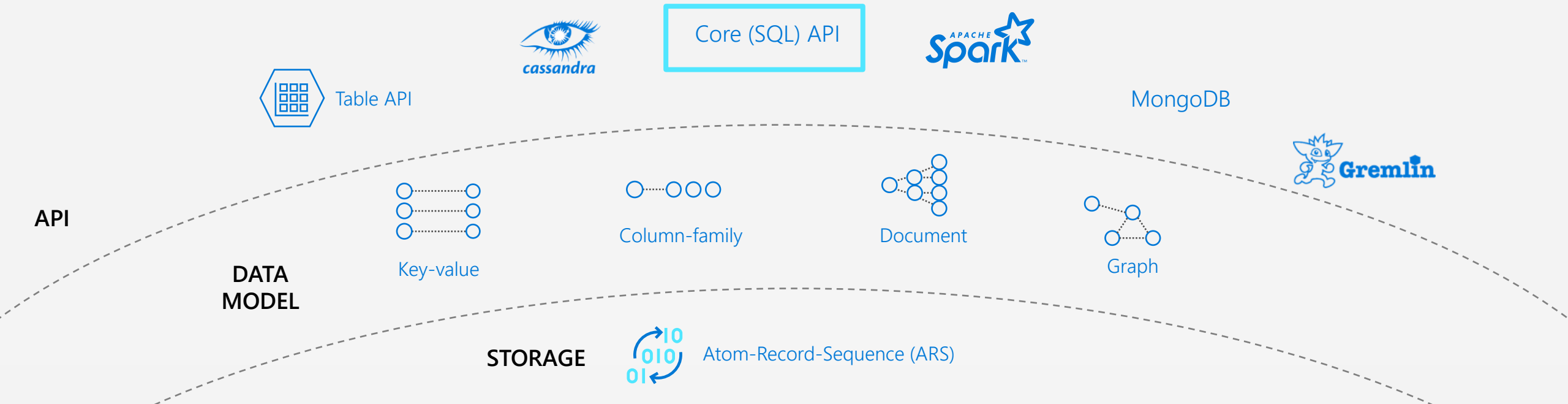




Choosing the right API

	Core (SQL)	MongoDB	Cassandra	Azure Table	Gremlin
New projects being created from scratch	✓				
Existing MongoDB, Cassandra, Table, or Gremlin data		✓	✓	✓	✓
Analysis of the relationships between data					✓
All other scenarios	✓				

- In general, the **Core (SQL) API** SDK and Developer Ecosystem/Documentation is most diverse – making it ideal for new projects
- **Cost** is associated with **Storage** + **Throughput** (see later), not choice of **API**
  - Although, choosing an API for data that's unsuitable for data model leads to more **Throughput** + **Storage** = **$$$** (e.g. using SQL queries to implement a node-edge data model – rather than Gremlin)



Azure Cosmos DB

Pricing formula

Provisioned throughput
Hourly provisioned Request Units (RU/s) normalized across reads, writes, queries, updates, and deletions.

+

Consumed storage
Hourly consumption of SSD-backed storage (GBs) for data and indexes.

=

Total price

Provisioned Throughput

Throughput	Provisioning (DB)	Writes (Acct)	Contract (Acct)
# of Request Units (RU/s)	**Container level** Assign throughput to specific database containers or **Database level** Share throughput across all database containers	**Single-master** Write data to a single Azure region or **Multi-master** Write data to any number of Azure regions	**Pay hourly** Monthly billing, based on hourly provisioned RU/s or **Reserved capacity** One- or three-year terms offer ≤65% savings

Consumed Storage

(**Data storage** + **Index storage**
Auto-indexing estimated at 30-50% of data.
Adjust index storage with custom policies.) × **# of Azure regions**

What are Request Units (RU)?

Request Units (RUs) is a **rate-based currency** – e.g. 1000 RU/second

Expressed in Request Units per second (RU/s)

- Represents the "cost" of a request in terms of CPU, memory and I/O

Performance can be provisioned:

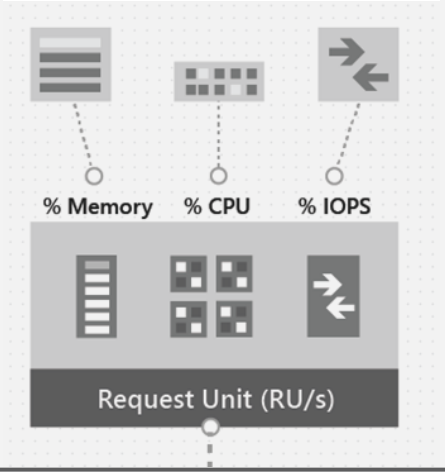
- at the **database**-level
- at the **collection**-level (aka **container**)
- or both

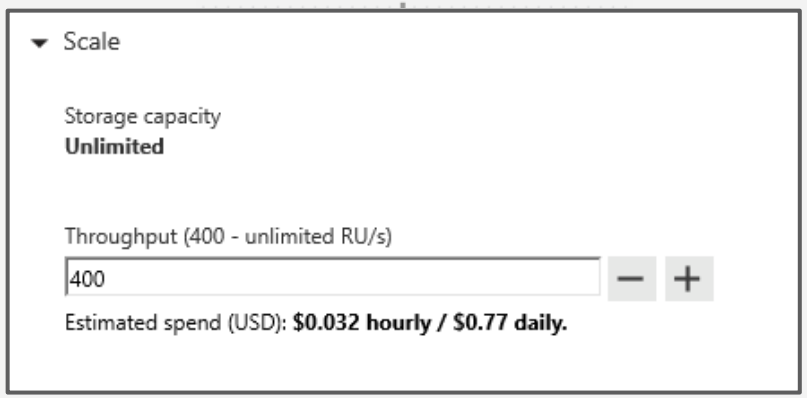
Can change RU/s:

- Manually from Azure Portal
- Programmatically with API calls
- Auto-scale to scale based on usage

"RU"

Abstracts physical resources for performing requests





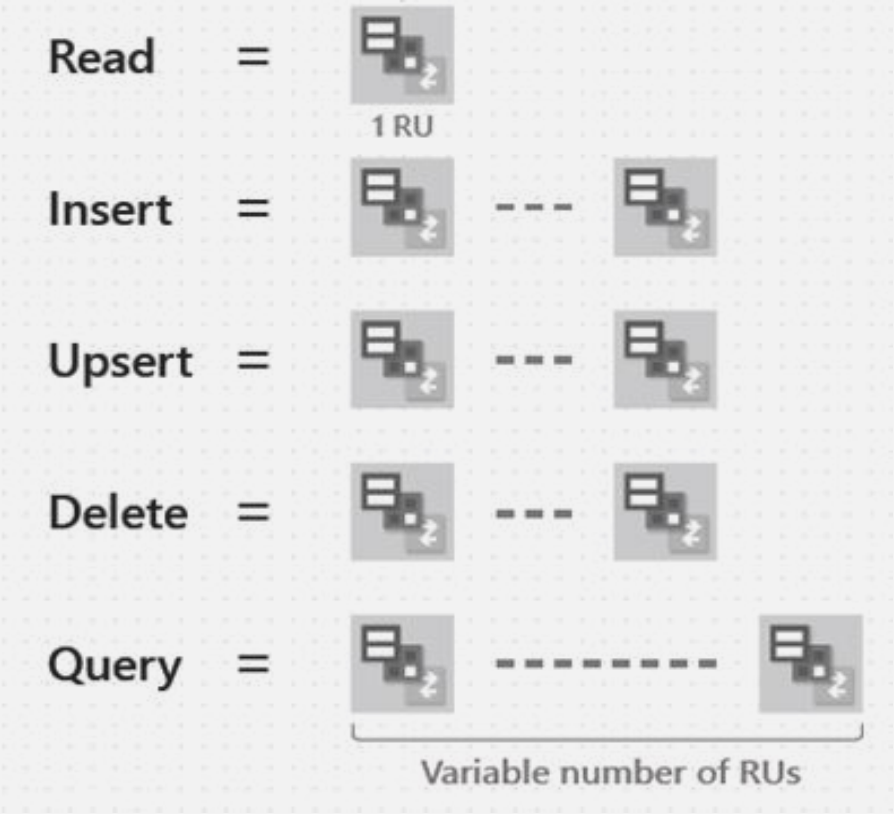
What are Request Units (RU)?

Each request consumes a defined # of RUs (given a certain setup)

- Approx. **1 RU** = **1** read of **1 KB** document
- Approx. **5 RU** = **1** write of a **1 KB** document

Query (deterministic - depends on query complexity & documents involved)

- Document size
- Partition scans
- Number of indexed fields
- Types of indexes
- Consistency model choice



Estimating required RUs

To do

- Identify query & access patterns
  e.g. **`Top 5 queries`**, **`# of Reads/writes per second`**
- Use '**`Request Charge`**' property from SDK + sample document to see # RU / operation
- POC / Load test -> Scale up, and scale down

Operation Type	# Requests per sec	# RU's per Request	RU's Needed
Write Single Document	10,000	10	100,000
Top Query #1	700	100	70,000
Top Query #2	200	100	20,000
Top Query #3	100	100	10,000
Total RU			200,000 RU

Estimating required RUs - example

Storage Cost

Avg Record Size (KB)	1
Number of Records	100,000,000
Total Storage (GB)	100
Monthly Cost per GB	$0.25
Expected Monthly Cost for Storage	$25.00

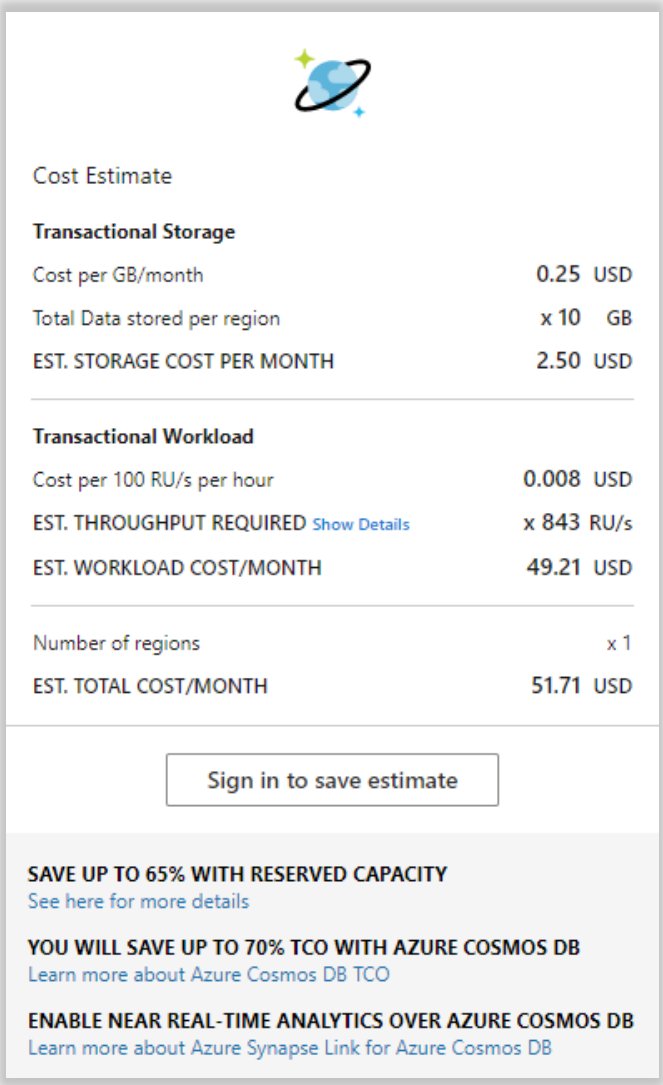
Throughput Cost

Operation Type	Requests / s	Avg RU / Request	RU Needed
Create	100	5	500
Read	400	1	400

Total RU	900
Hourly Cost / 100 RU	$0.008
Monthly Cost /100 RU	$6.00
Expected Monthly Cost for Throughput	$54.00

Total Monthly Cost

```
[Total Monthly Cost] = [Monthly Cost for Storage] + [Monthly Cost for Throughput]
                     =   $25                      +   $54
                     =   $79/month
```



https://cosmos.azure.com/capacitycalculator/

Database VS Container Level Throughput

In general, container level throughput is a good choice for Production workloads

- Predictable performance since each container is guaranteed its provisioned RU's

Choosing Database Level Throughput can also be a good option if:

- We are migrating **many** containers in a lift-and-shift migration (from Table Storage, MongoDB, or Cassandra) and do not know how much throughput to set for each one
- Multi-tenant applications where each LOB/user is represented by a separate container

Realistically – leverage both case by case

Minimum Provisioned Throughput limits

The minimum provisioned throughput:

On any container is `400 RU`

Once a container is storing data:

there is a throughput minimum of `10 RU/GB`

Once a container has provisioned `X RU`:

the future minimum throughput is `X RU / 100`

```
MAX
(
    400 RU
    Current storage * 10 RU/GB
    Max Throughput provisioned / 100
)
```

Minimum RU/s

Example:

- You create a new, empty container. The minimum RU/s you may provision is `400 RU`.
- You then ingest `50 GB` of data to the container. The minimum RU/s you may provision is `50 GB * 10 RU/GB = 500 RU`.
- You provision `1,000,000 RU` on the container during a migration. `10,000 RU` is now the minimum throughput.

Choose the Azure Cosmos DB model that suits your needs

Azure Cosmos DB offers three models for database operations suited for different needs and use cases. To select the right model for you, consider the following criteria:

		Standard provisioned throughput	Auto-scale provisioned throughput	Serverless
1	**Workload size** Operational requirements	**Unlimited** Supports high-throughput workloads at any scale.	**Unlimited** Supports high-throughput workloads at any scale.	**Moderate** Supports moderate bursts up to 5k-20K RU/s and 1TB of data.
2	**Performance needs** Speed performance, and availability requirements	**Very high** Guarantees <10 ms latency and 99.999% availability at any scale, across all Azure regions	**Very high** Guarantees <10 ms latency and 99.999% availability at any scale, across all Azure regions	**Moderate** Limited to single Azure region deployments in preview.
3	**Data patterns** Predictability and consistency of workload usage	**Highly predictable** Suited for predictable and consistent workloads, or where direct throughput management is desired.	**Less predictable** Suited for less predictable workloads, and when direct throughput management overhead isn't desired.	**Unpredictable** Ideal for spiky workloads with long idle periods and sporadic requests.
	Billing details	Reserves capacity for your workloads, which is billed by the hour. You pay for what you provision, starting from 400 request units per second (RU/s).	Sets a custom throughput maximum and is billed by the hour. You pay for what is used, starting at 400 RU/s or 10% of maximum.	Billed by request unit (RU), for the total consumed per operation, starting at zero.

Standard provisioned throughput

Unlimited

Supports **high-throughput** workloads at any scale.

Very high

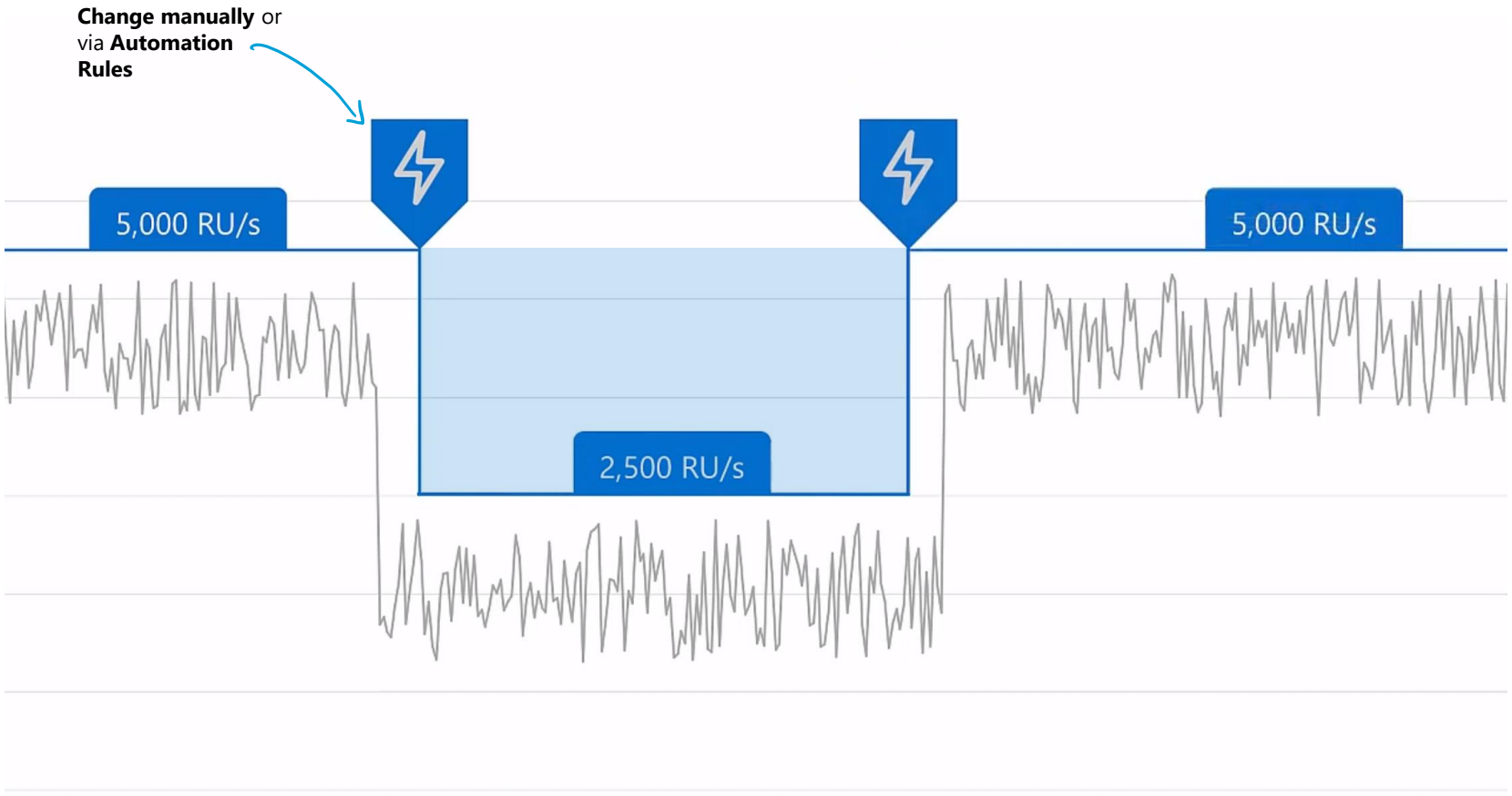
Guarantees <10 ms latency and 99.999% availability at any scale, across all Azure regions

Highly predictable

Suited for **predictable** and **consistent** workloads, or where direct throughput **management is desired**.

Billing

Reserves capacity for your workloads, which is billed by the hour. You pay for what you provision, starting from 400 request units per second (RU/s).



Autoscale provisioned throughput

Unlimited

Supports **high-throughput** workloads at any scale.

Very high

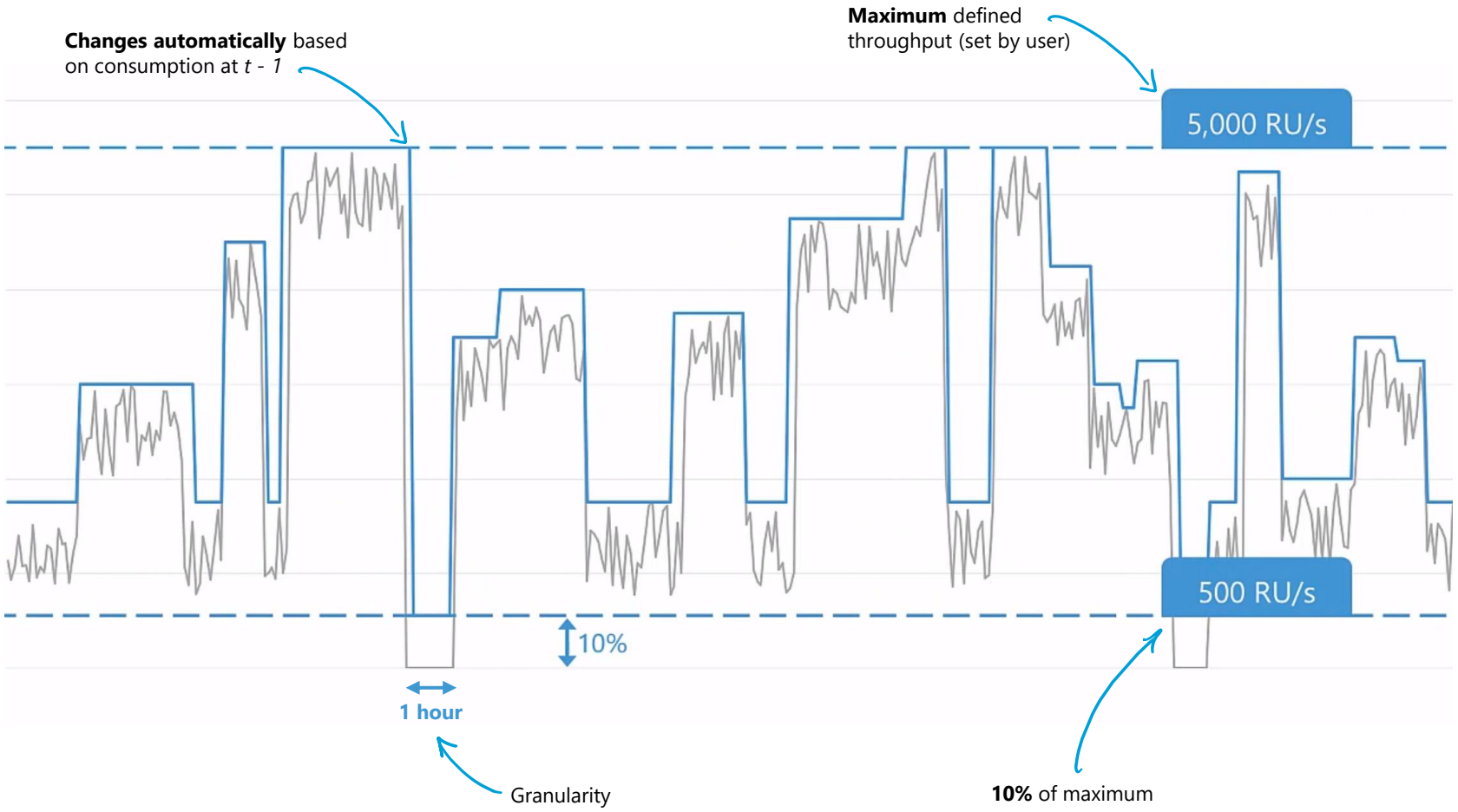
Guarantees <10 ms latency and 99.999% availability at any scale, across all Azure regions

Less predictable

Suited for **less predictable** workloads, and when direct throughput **management** overhead **isn't desired**.

Billing

Sets a custom throughput maximum and is billed by the hour. You pay for what is used, starting at 400 RU/s or 10% of maximum.



Serverless **(preview)**

Moderate

Supports **moderate bursts up to 5K-20K RU/s**.

Moderate

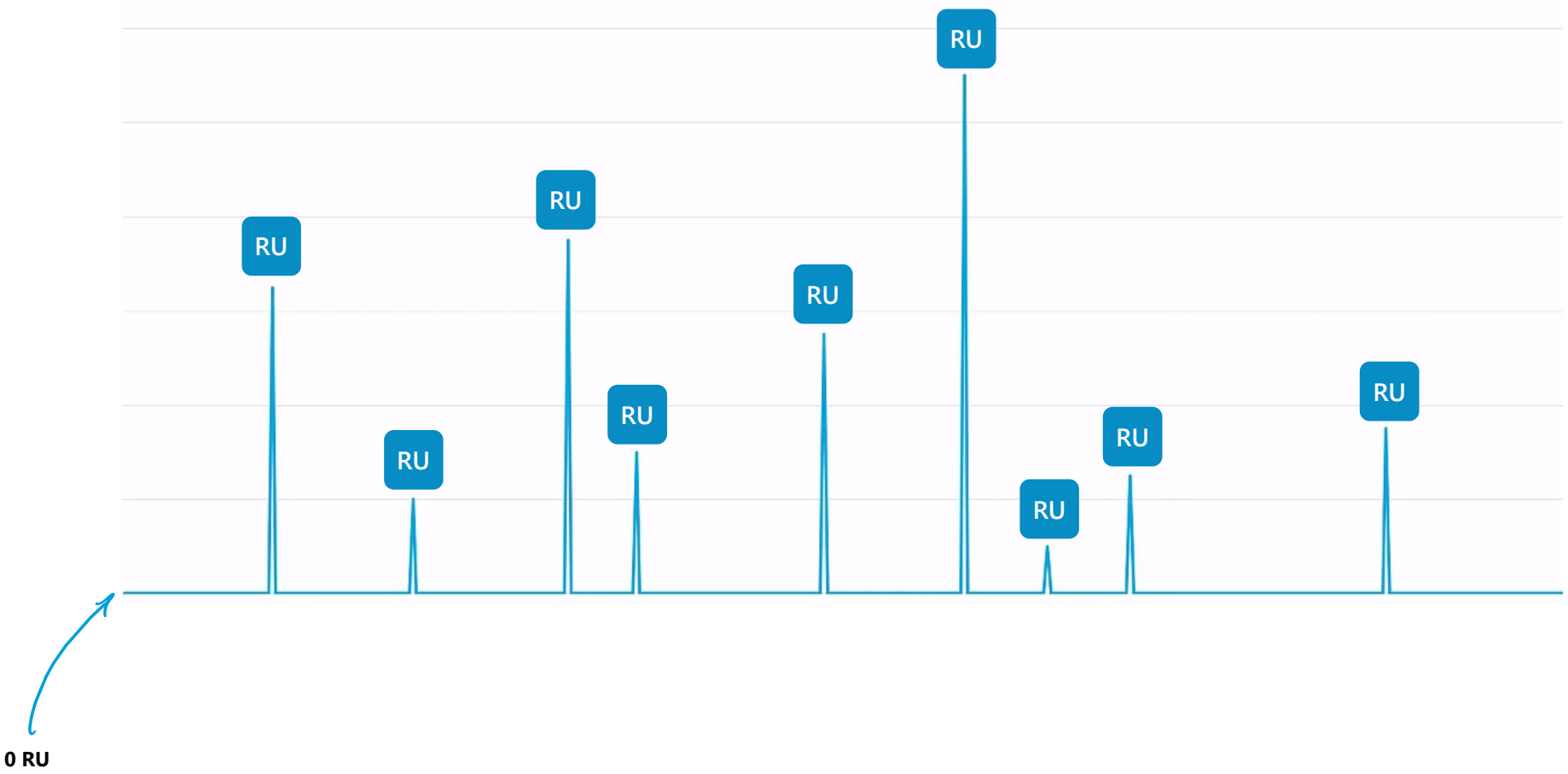
Limited to single Azure region deployments in preview.

Unpredictable

Ideal for **spiky workloads** with **long idle periods** and sporadic requests.

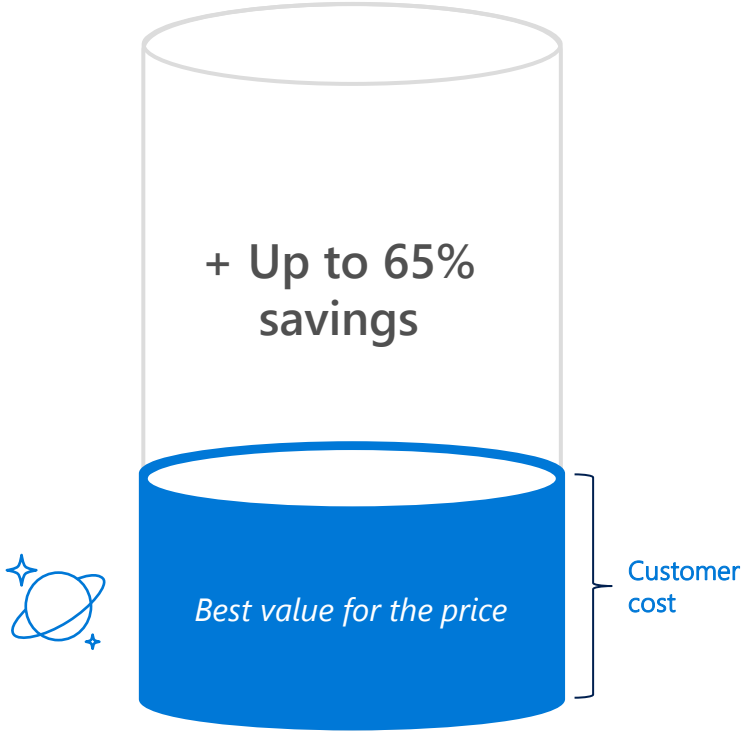
Billing

Billed by request unit (RU), for the total consumed per operation, starting at zero.



Cosmos DB reserved Capacity can provide up to 65% savings

Save up to 65% with Azure Cosmos DB reserved capacity pricing



Best elasticity for the price

Saturation of provisioned capacity via sub-core multiplexing

Deep integration w/ Azure Networking

No need to segment into read and write workloads – w/ unified and normalized throughput currency - RUs

Total Time To Live (TTL) is free

Enterprise-ready (security, compliance, encryption) at no additional cost

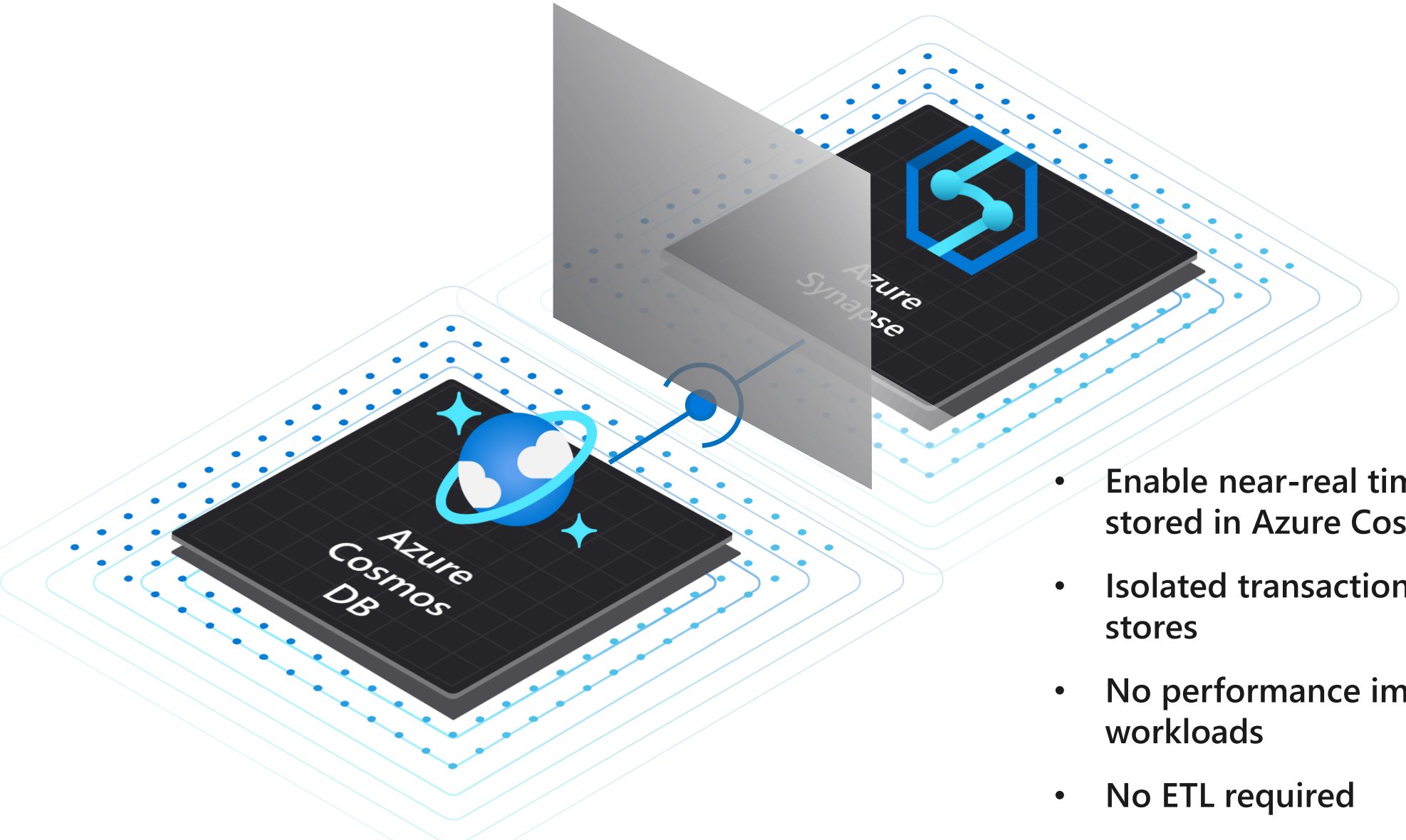
Microsoft Azure

Azure Synapse Link

Run near-real time analytics with cloud-native HTAP for Azure Cosmos DB

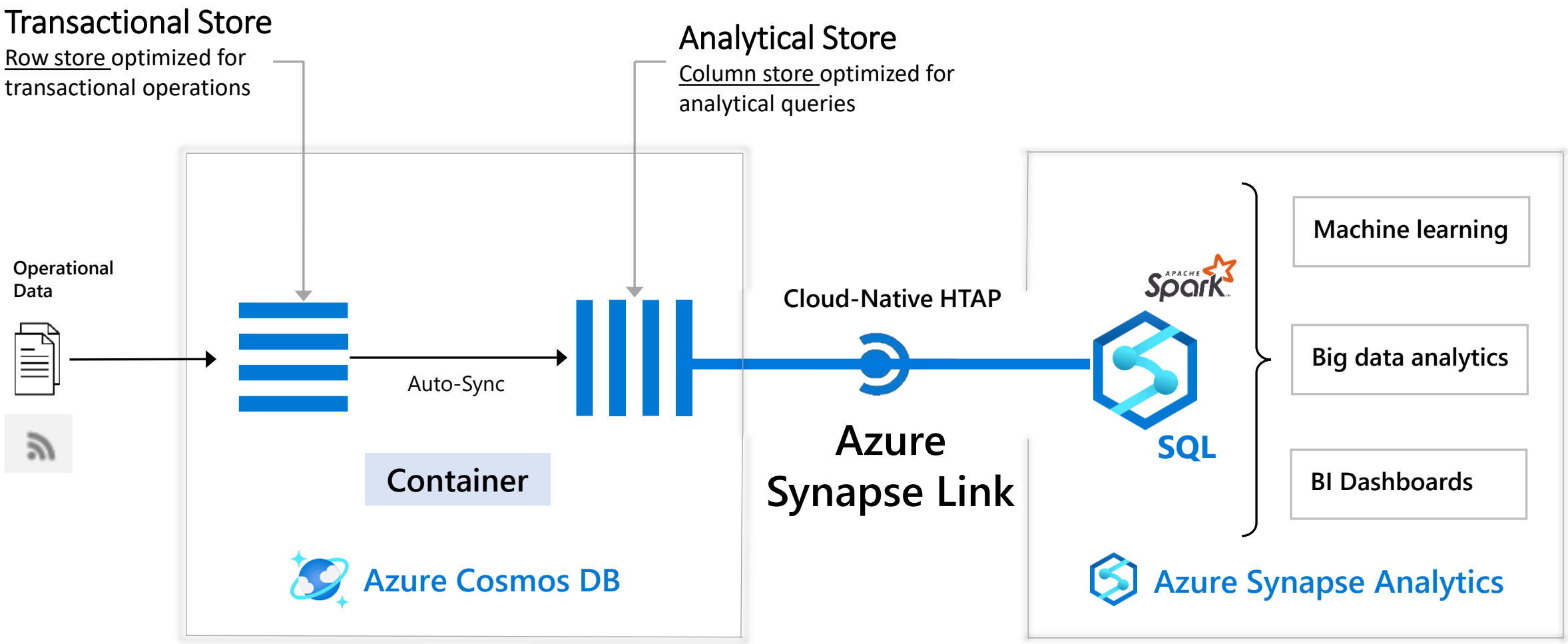
Azure Synapse Link for Azure Cosmos DB

Near-real time analytics over transactional data



- Enable near-real time analytics over data stored in Azure Cosmos DB with a click
- Isolated transactional and analytical data stores
- No performance impact on transactional workloads
- No ETL required

How Azure Synapse Link (preview) works



Generate near real-time insights on your operational data

Samples repo: https://aka.ms/cosmosdb-synapselink-samples

Questions?