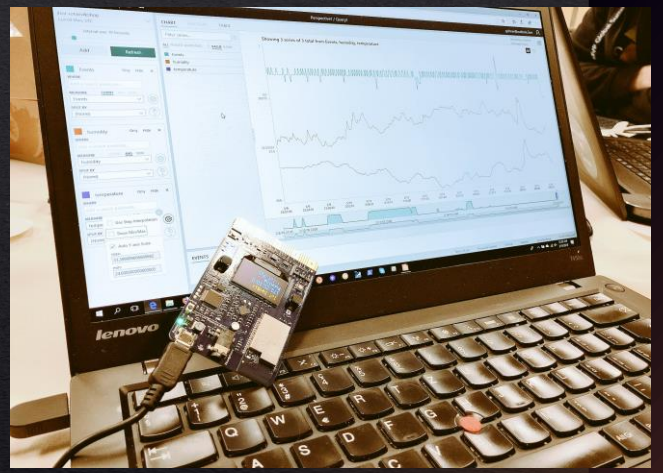
Build and Connect to Database using Azure Cosmos DB

Sajeetharan Sinnathurai

Program Manager, Azure Cosmos DB

@kokkisajee

Love at First Code: My Journey with Azure Cosmos DB as a Dev



Azure Cosmos DB was launched on
May 10, 2017

Bookmarks
@kokkisajee

rimma nehme @rimmanehme · Mar 5, 2018
At the #MVPSummit today, we've auctioned off an #Azure #CosmosDB poster with the signatures of the #CosmosDB team for 54 #CosmosDB blogs in 12 months. That's a lot of #CosmosDB ❤️ and dedication. Look forward to it @kokkisajee! Thank you. #mvpbuzz #MVPSummit2018

> **Sajeetharan** @kokkisajee · Mar 5, 2018
> I will write 54 blogs on #CosmosDB #azure and please send a similar flyer signed by engineers @rimmanehme 😎

1 15 40

Agenda

- Why Azure Cosmos DB for developers?
- How to talk to Azure Cosmos DB?
- Cosmos DB NOSQL API JS SDK 4.0.0
- Project walkthrough – Contoso real estate
- QnA

Azure Cosmos DB – next generation database for modern apps

Guaranteed performance

Just **set the throughput** your need
SLA guaranteed throughout and latency
Speed layer

Operational + Analytical

OLTP +OLAP with the same DB
Real-time Analytics and AI

24/7 Uptime

High Availability **99.999%**
Worldwide data **replication**
with 1 click (if you need it)

Flexibility

3 **operational modes** to adapt to different business requirements
5 **consistency models**
Flexible **data model** (JSON)

Fully Managed

No need to manage nodes
Automatic backups/failover
Managed data replication

Elastic scale out of storage & throughput

Design philosophy based on response times
Adds more capacity as you need it

Versatility

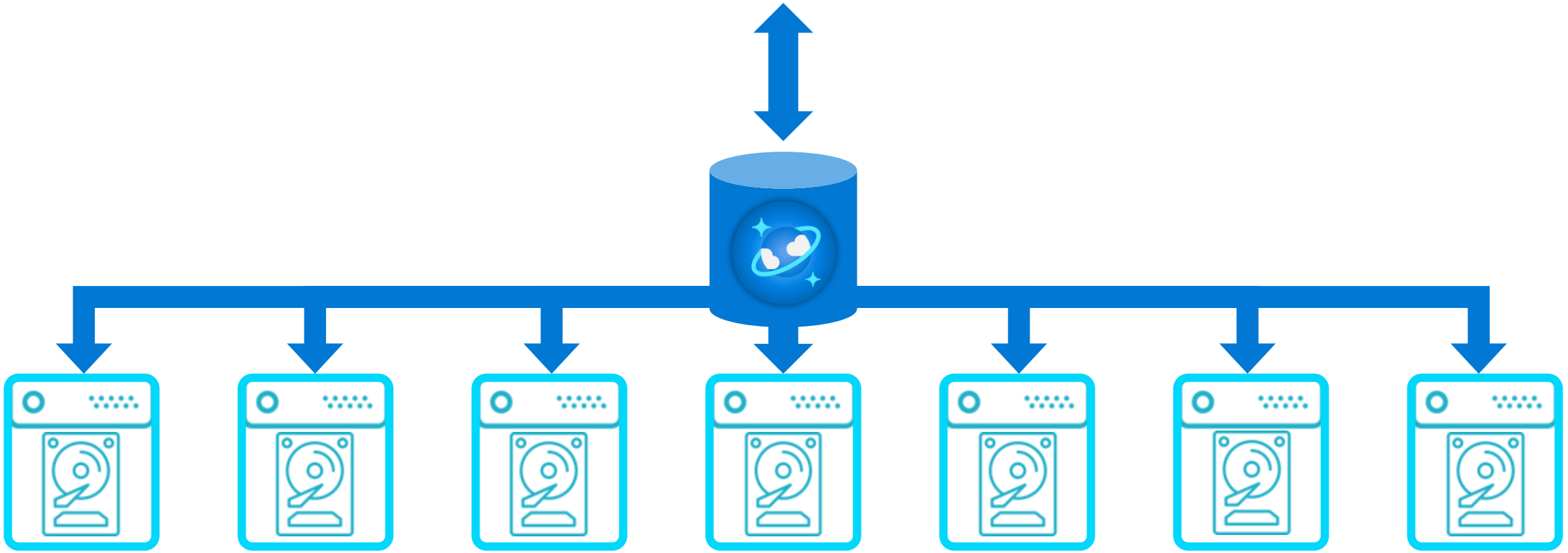
Supports both **relational and non-relational** workloads
Any API and SDK
Leverage your assets & knowledge

Automation

Trigger business processes as data changes

What is Azure Cosmos DB?

horizontally scalable

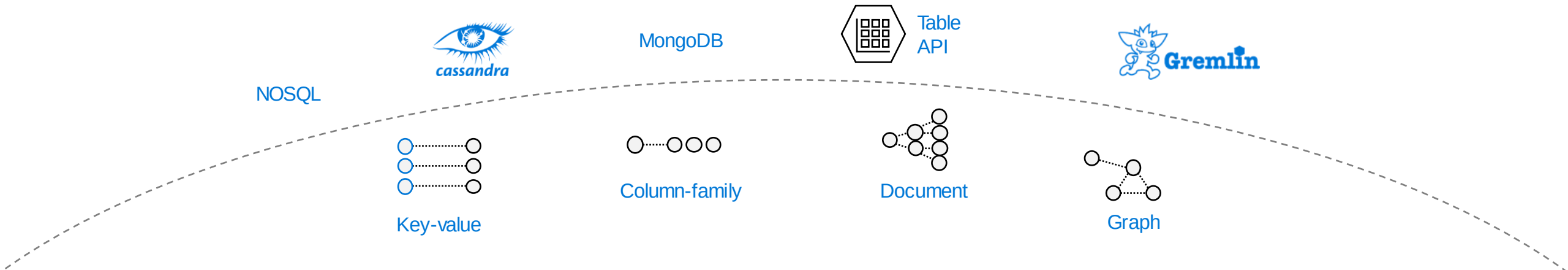


Unlimited throughput
Unlimited storage capacity

Multiple Data Models and API's

Use the model that fits your requirements, and the APIs, tools, and frameworks you prefer

- Cosmos DB offers a multitude of APIs to access and query data including, SQL, various popular OSS APIs, and native support for NoSQL workloads.
- Use key-value, columnar, graph, and document data
- Data is automatically indexed, with no schema or secondary indexes required
- Blazing fast queries with no lag



Various Pricing and Scalability Models To Meet Your Needs

- Free local emulator
- Free Tier
- Serverless
- Standard Provisioned Throughput
- Autoscale Provisioned Throughput

[Azure Cosmos DB Capacity Calculator]

Question time : How long you've been using Azure Cosmos DB?

When can Azure Cosmos DB help?

Do you have applications that **struggle to scale with demand**?

Do you have processes that need **low-latency guarantees**?

Do you have applications where **high availability is a critical requirement**?

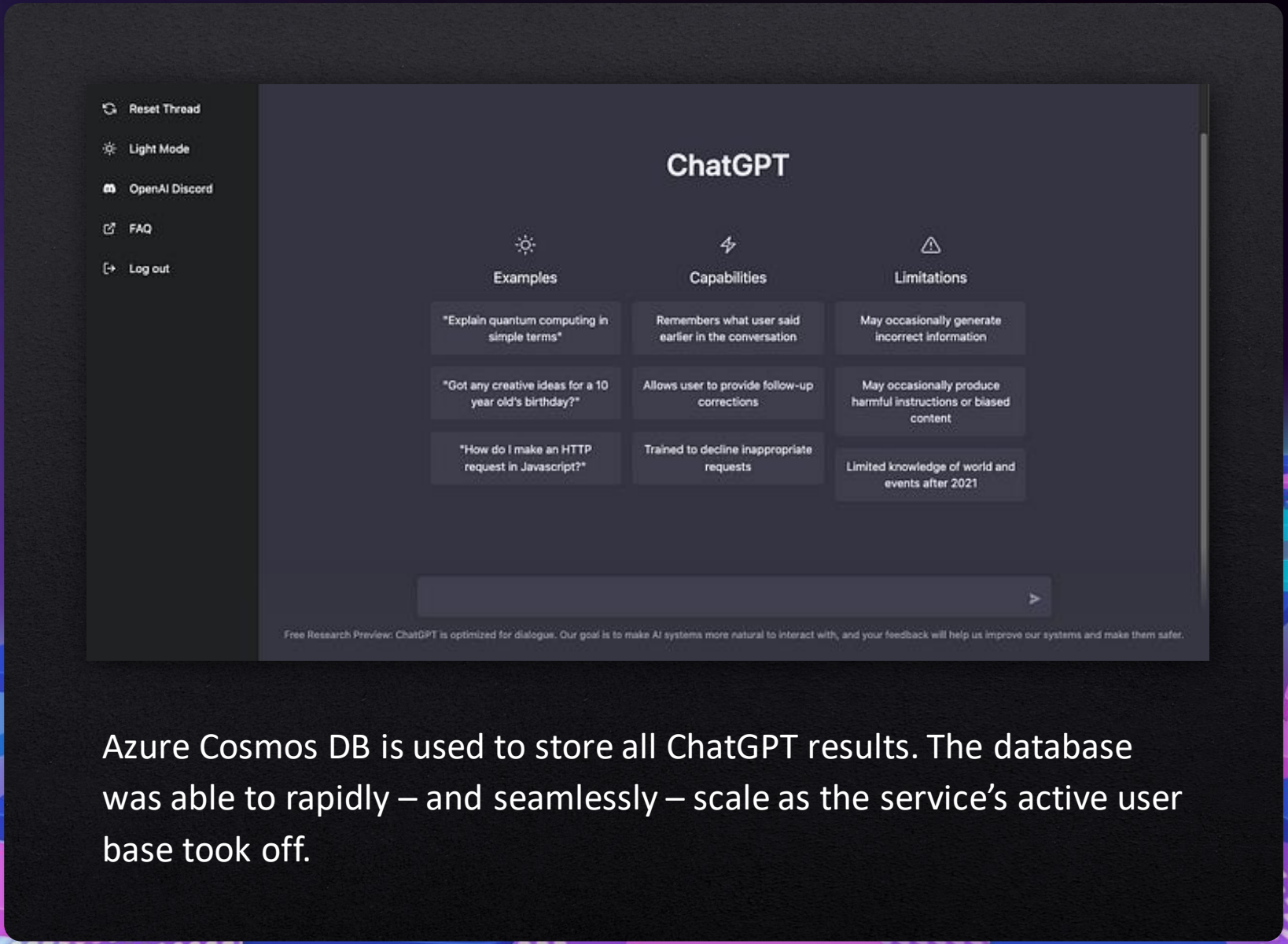
Do you have applications that require **geo-replication to serve data anywhere**?

Do you have applications whose **data model changes** rather often or with **heterogeneous** data models?

Are you thinking about **event-driven or microservices architectures**?

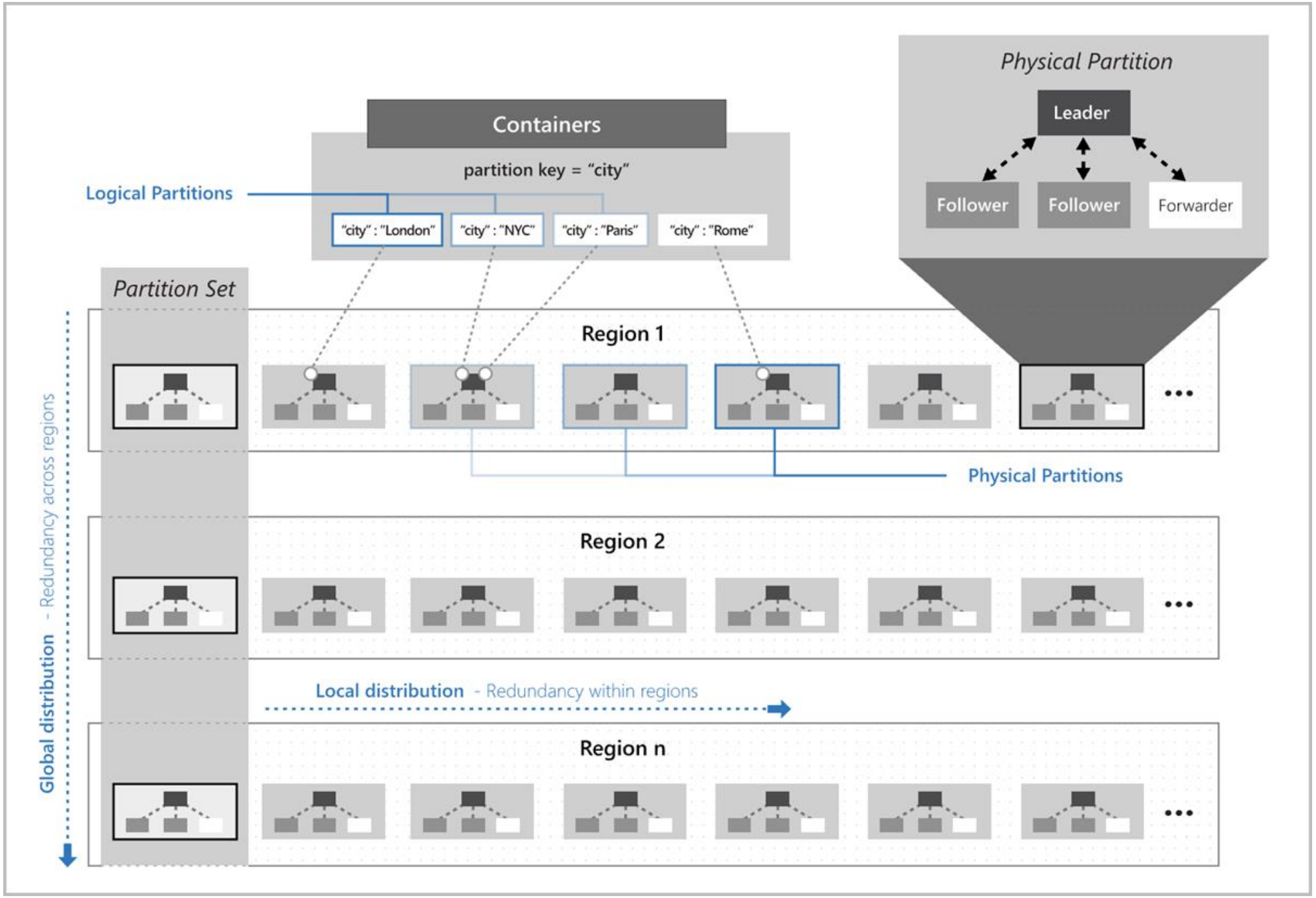
Any **yes**?

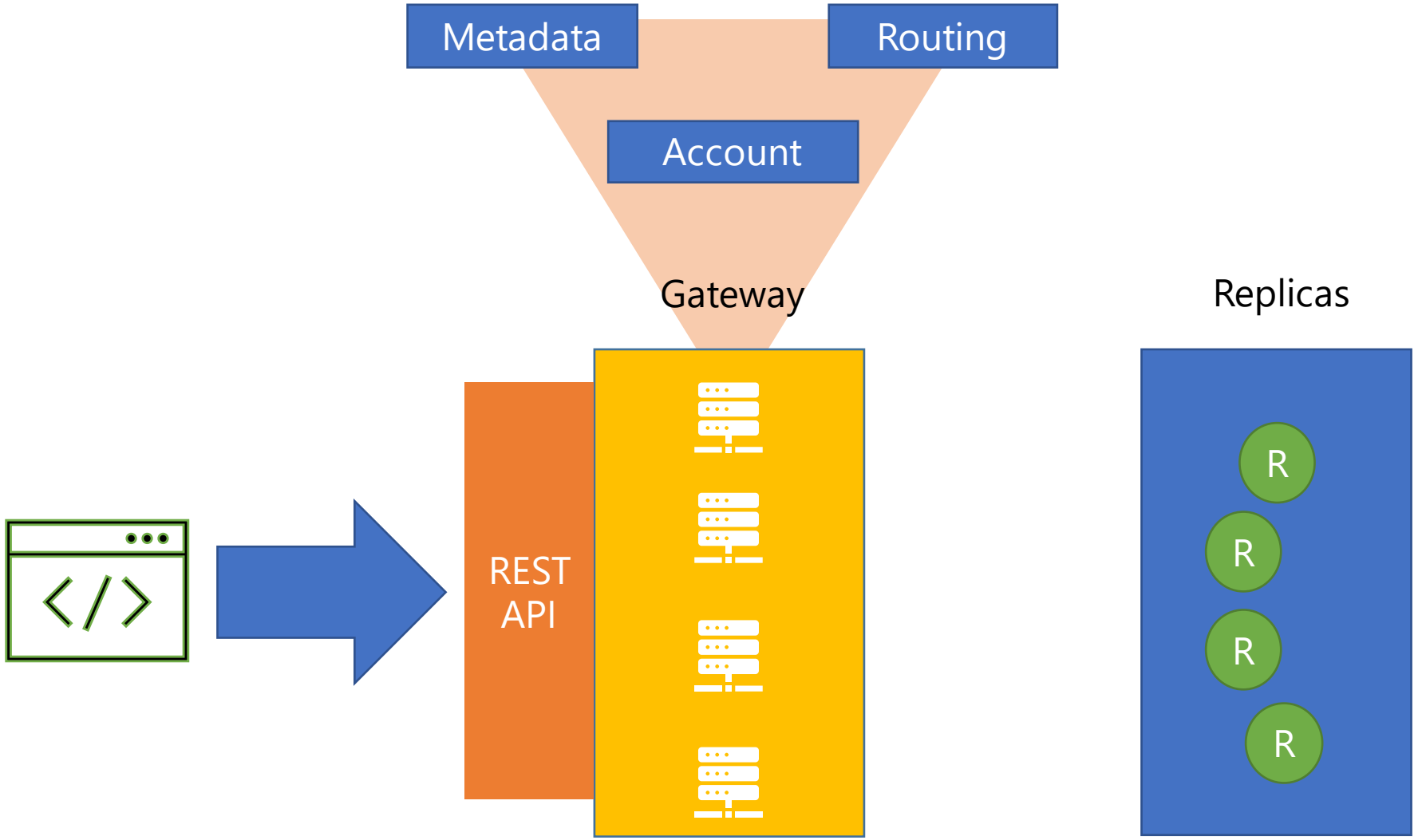
OpenAI



Azure Cosmos DB is used to store all ChatGPT results. The database was able to rapidly – and seamlessly – scale as the service's active user base took off.

How to talk to Cosmos DB with code?


Physical Partition
Leader
Follower
Follower
Forwarder
Containers
partition key = "city"
Logical Partitions
"city" : "London"
"city" : "NYC"
"city" : "Paris"
"city" : "Rome"
Partition Set
Region 1
Physical Partitions
Region 2
Local distribution - Redundancy within regions
Region n
Global distribution - Redundancy across regions


Metadata
Routing
Account
Gateway
Replicas
REST
API
R
R
R
R

How can we use Azure Cosmos DB in our code/apps?

- APIS (REST)
- SDKs
- ORMs
- Data API Builder (GraphQL) 
- Drivers

Azure Cosmos DB *Data Plane* REST API

- Reference: https://docs.microsoft.com/rest/api/cosmos-db/
- Work with resources ***in*** a Cosmos DB account
  - Databases, Containers, Documents, Sprocs, UDFs, Documents, Queries
- Can NOT create/manage Cosmos DB accounts or keys with this API
- NOSQL API only! (Against other APIs = 404/failure)
- Use the account endpoint: https://youraccount.documents.azure.com

Azure Cosmos DB *Management* REST API

Reference: https://docs.microsoft.com/rest/api/cosmos-db-resource-provider/

Work with Azure Cosmos DB accounts and keys

For other tasks, use the REST API (previous slide)

All Cosmos DB APIs supported

Endpoint: the Azure management API itself

Other layers (SDKs, Azure CLI, Azure Portal etc.) use this RP API

Azure Cosmos DB Language Support

In theory, you can develop in any *language*! REST / Graphql

But… in reality, this decision process makes sense:

- *Which Cosmos DB API shall we use?*
  - Compatibility with any existing codebase? (Mongo DB, Cassandra…)
  - + Other considerations: skillset, hybrid platform?
- *How can we develop for that Cosmos DB API?*
  - SDKs provided by Azure Cosmos DB - .NET, Java, Python, Node.js
  - For Mongo DB/Cassandra/Graph APIs… use that community/platform's existing tools

Azure Cosmos DB (NOSQL API) Language Support

.NET SDKs (Framework, Core, Standard)

- V2: still supported but do not use for new work
- V3: current release – notably, includes bulk support which was separate in V2
- V4: preview

Java SDKs

- V2: sync and async versions – still supported but do not use for new work
- V4: current version

Node.js SDK (3.17.3)

Python SDK (4.3.0)

GoLang (0.3.4)

Azure Cosmos DB Language Support

Mongo API

-Official drivers provided by MongoDB

-13 officially supported libraries including .NET/C#, Java, Go, Node.js, Python, Rust

- CData drivers – specifically for Cosmos DB accounts with MongoDB API

- Mongoose – Node.js object-database mapper (see how-to guide)

The idea: Cosmos DB MongoDB API account can be substituted for a MongoDB instance – change nothing in your code

- Not all Mongo commands are supported; review and test early!!

Cassandra API

- Cassandra drivers from Datastax and gocql
  - Datastax is commercial Cassandra vendor; has Azure MP offerings
  - Java, .NET/C#, Node.js, Python, C++, PHP
  - Gocql is OSS: Cassandra client for Go
- Spring Data
  - Spring is a major Java framework
- Carefully review supported Cassandra data types, CQL functions, and limits!
- Also review differences

Gremlin API

Gremlin API based on Apache Tinkerpop graph database standard.

Uses Gremlin query language to interact with data.

Use Gremlin API for compatibility, or to implement a graph-capable application.

Use a compatible client library from Apache or a third party. (we recommend drivers supported by Apache Tinkerpop.) Can also use Spring Data.

Careful! Review Cosmos DB Gremlin compatibility and limits!

Table API

Table API is for apps written to use Azure Table Storage.

SDKs for .NET/C#, Java, Python, and Node.js. The Azure Storage REST API for Table is also supported.

Recommended to use the Cosmos DB-specific SDKs rather than Azure Storage SDKs.

Carefully review the differences between Azure Table Storage and Cosmos DB Table API, as well as the benefits of Cosmos DB Table API.

Azure Cosmos DB NOSQL JS SDK

- Offers a powerful and flexible solution for developing Node.js applications that leverages the capabilities of Azure Cosmos DB

- Provides a rich set of APIs and Features to perform operations and enable local development experience

- Latest version [@azure/cosmos - npm (npmjs.com)](https://www.npmjs.com/package/@azure/cosmos) 3.17.3

https://aka.ms/cosmosdb-js-sdk

Azure Cosmos DB V4.0.0 Private preview

- ## Client-Side Request Diagnostics

  Includes all service interactions, time spent on the SDK (serialization or initialization), server-side latency, and connectivity troubleshooting.

- ## Change Feed support

  Users can now consume change feed at their own pace, choosing to fetch change feed for an entire container, specific feed range, or a partition key.

- ## Hierarchical Partition Key & Index Metrics

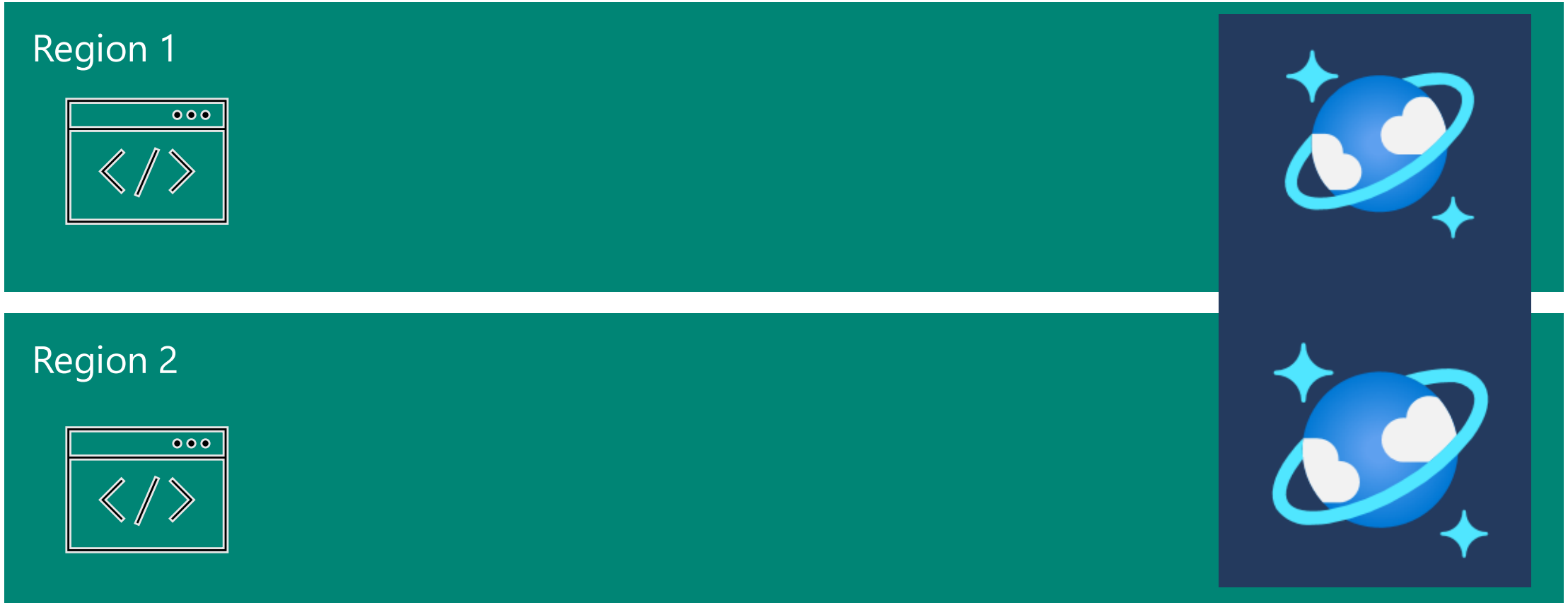
  You can configure up to a three-level hierarchy for your partition keys to further optimize data distribution and for a higher level of scaling

Javascript SDK Client side request Diagnostics

```
const clientOps: CosmosClientOptions = {
  endpoint: "",
  diagnosticLevel: DiagnosticLevel.debugUnsafe
};

client = new CosmosClient(clientOps);
```

High latency – regional preference



High latency – regional preference

Regional preference

Enable Preffered Regions

.JS

```
const { CosmosClient } = require("@azure/cosmos");

const options = {
  applicationName: "MyApp",
  preferredRegions: ["West US", "Central US"], // You can specify multiple region
s in an array
};

const client = new CosmosClient({
  endpoint: "your-endpoint",
  key: "your-key",
  ...options,
});
```

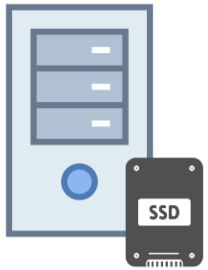
Cosmos DB for MongoDB

Use Azure Cosmos DB as a Mongo DB database, without having to manage the database infrastructure

Compatible with MongoDB Apps/Tooling/Drivers/SDKs

Two architectures

Request Unit (RU)



vCore

Native Azure service integrations

https://aka.ms/cosmos-mongo-bookstore

Cosmos DB under the hood

Your client

Mongo Wire Protocol

Cosmos DB

Gateway can handle multiple versions, **same code base** – upgrades are config, not bits

Mongo Gateway

Internal protocol

No need to pay extra for a metadata replica for sharding

Throughput modes for different workloads

Watch the video

- Pay only for the throughput you need
- No warm-up period needed for scale-up
- Can scale in increments as small as 1/100th of a VM

Provisioned

- Fixed requirements

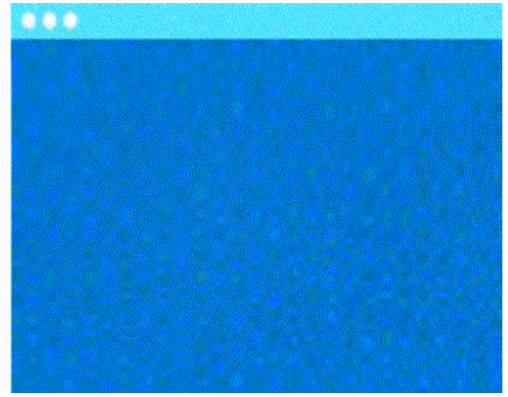
Autoscale

- Granularly scale from tens to millions of requests/sec

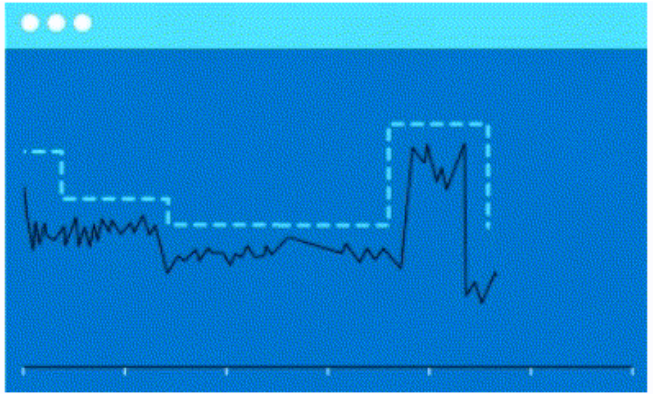
Serverless

- Pay for requests you use
- Suitable for spiky workloads

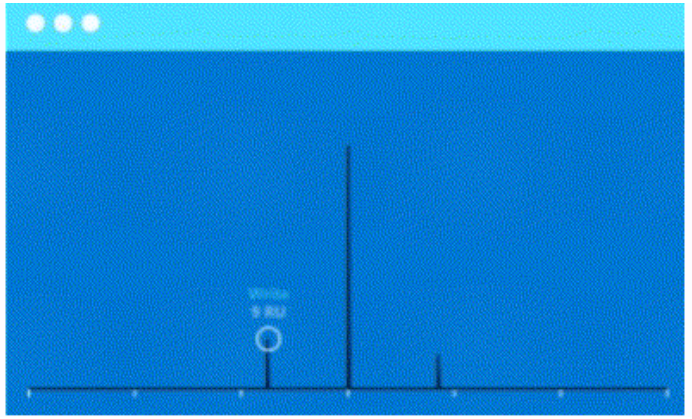
Provisioned
Predictable



Autoscale
Unpredictable traffic



Serverless
Mostly idle and occasional spike



Simplified app development

- Supports Mongo API wire protocol versions
  - 4.2
  - 4.0
  - 3.6
  - 3.2
- Leverage existing MongoDB drivers, libraries and tools
- Built-in MongoDB shell
- ACID-compliant multi-document transaction support with API v4.2
- Deep integration with key Azure services including Azure Synapse Analytics
- Change stream: Track and manage changes to database containers
- Emulator for local development

MongoDB Drivers

C | C++ | C# | Go | Java | Node.js | PHP | Python | Ruby | Rust | Scala | Swift

MongoDB Tools

Mongo Shell | Compass | Studio 3T | Mongoose | Robo 3T

Two Architectures

Request Unit (RU)

-Autoscale
-Provisioned
-Serverless

Pay for throughput and consumed storage

vCore

Pay for instances and disks. High availability optional.

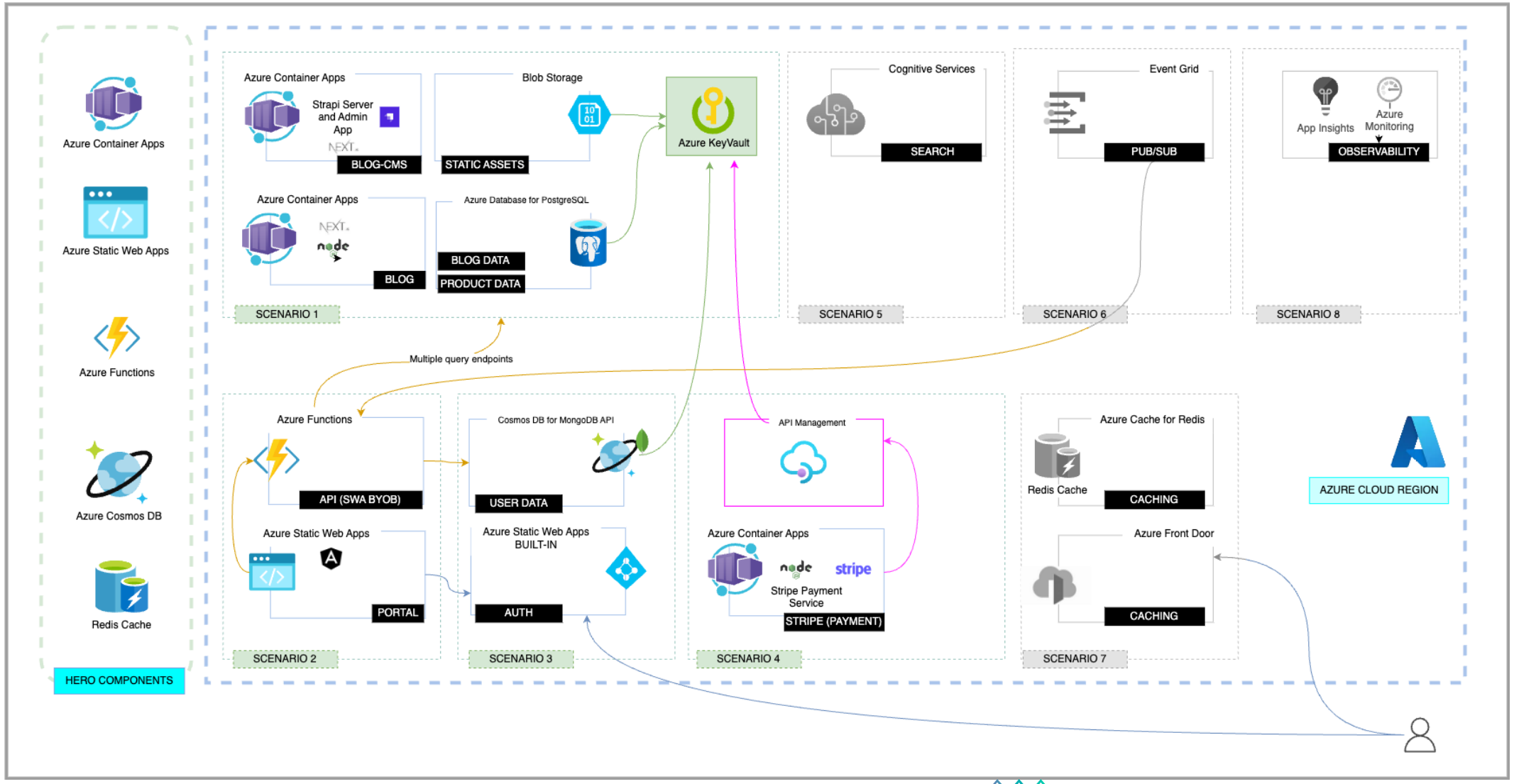
Code **walkthrough**

Provisioning CosmosDB
For Mongo DB

How to connect and query

Code to Azure: Testing Composable Apps

Think about: Microservices, API-First, Cloud-native, Micro-frontends

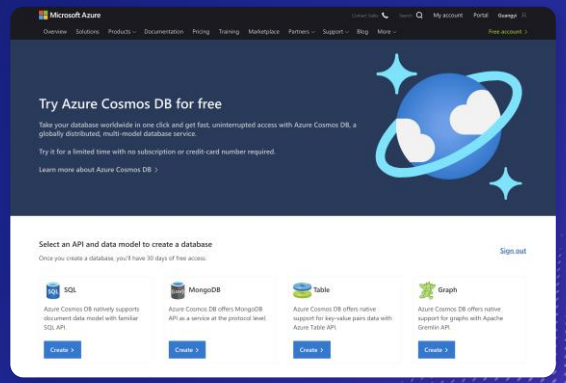


https://aka.ms/contoso-real-estate/github

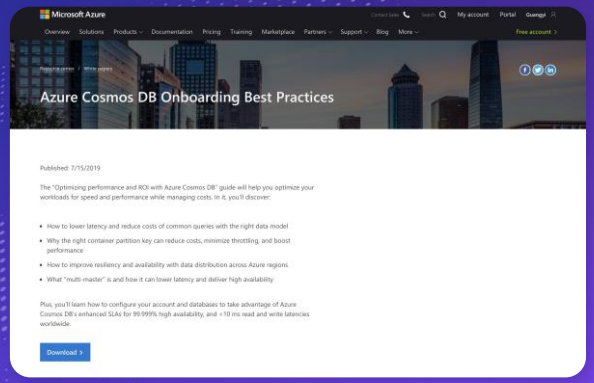
Any questions?

Are developers ready to build intelligent apps with Azure Cosmos DB?

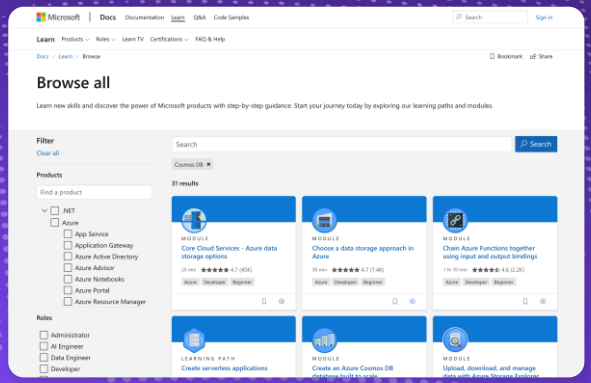
They can get started today:



[Try Azure Cosmos DB](#) free



[Learn how](#) to use Azure Cosmos DB through tutorials and best practices



[Build your skills](#) with Azure Cosmos DB learning modules

Jump into Azure Cosmos DB

Web AzureCosmosDB.com

GearUp aka.ms/AzureCosmosDBBoM

Social

 @AzureCosmosDB

 @AzureCosmosDB

Blog

 devblogs.microsoft.com/CosmosDB

 @kokkisajee