Azure Cosmos DB

The Swiss Army NoSQL Cloud Database

codit

Azure Lowlands

Thank you to our sponsors!

Gold Sponsors

INSPARK

betabit {...}

codit

Motion10

macaw

Silver Sponsors

winvision.

aims

Community Sponsors

Microsoft Integration User Group
The Netherlands

AZURE IOT
COMMUNITY

WAZUG.NL
AZURE USER GROUP NL

Experts Live Netherlands

What can you expect?

- Introduction of Cosmos DB
- Scenarios
- Cosmos DB Key Characteristics
- Multi-model
- Reference case
- Demos

Cosmos DB

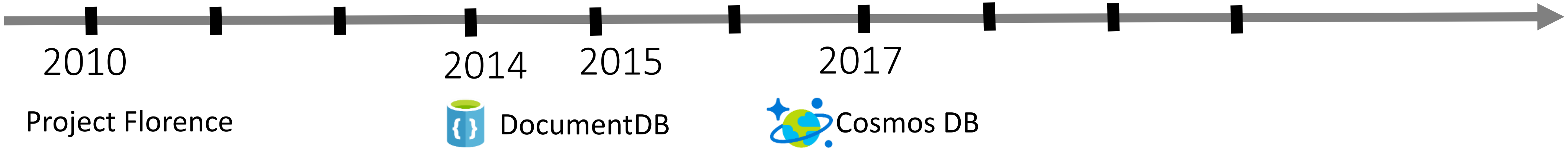
Introduction

Scenarios

Azure Cosmos DB Evolution

- Originally started to address the problems faced by large scale apps inside Microsoft

- Built from the ground up for the cloud
- Used extensively inside Microsoft
- One of the fastest growing services on Azure



2010

Project Florence

2014

DocumentDB

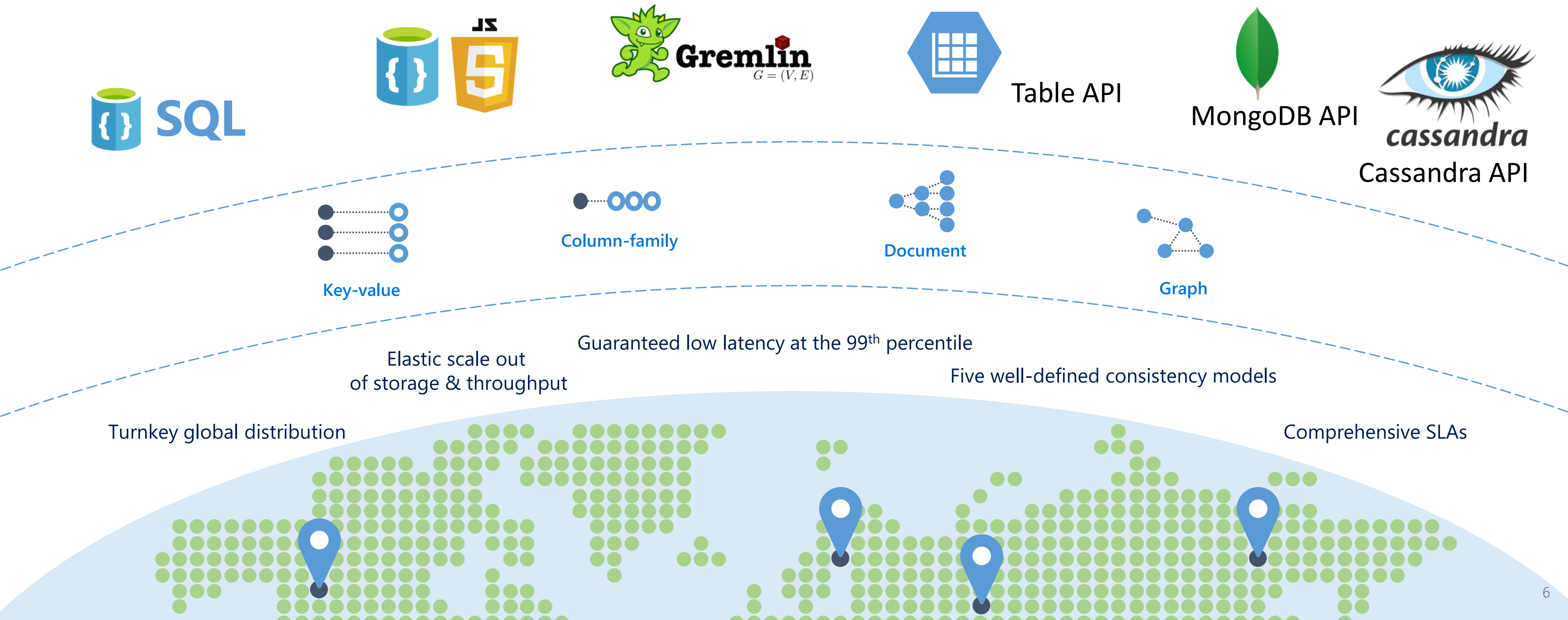
2015

2017

Cosmos DB

Cosmos DB Service

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



SQL

Gremlin $G = (V, E)$

Table API

MongoDB API

cassandra

Cassandra API

Key-value

Column-family

Document

Graph

Guaranteed low latency at the 99th percentile

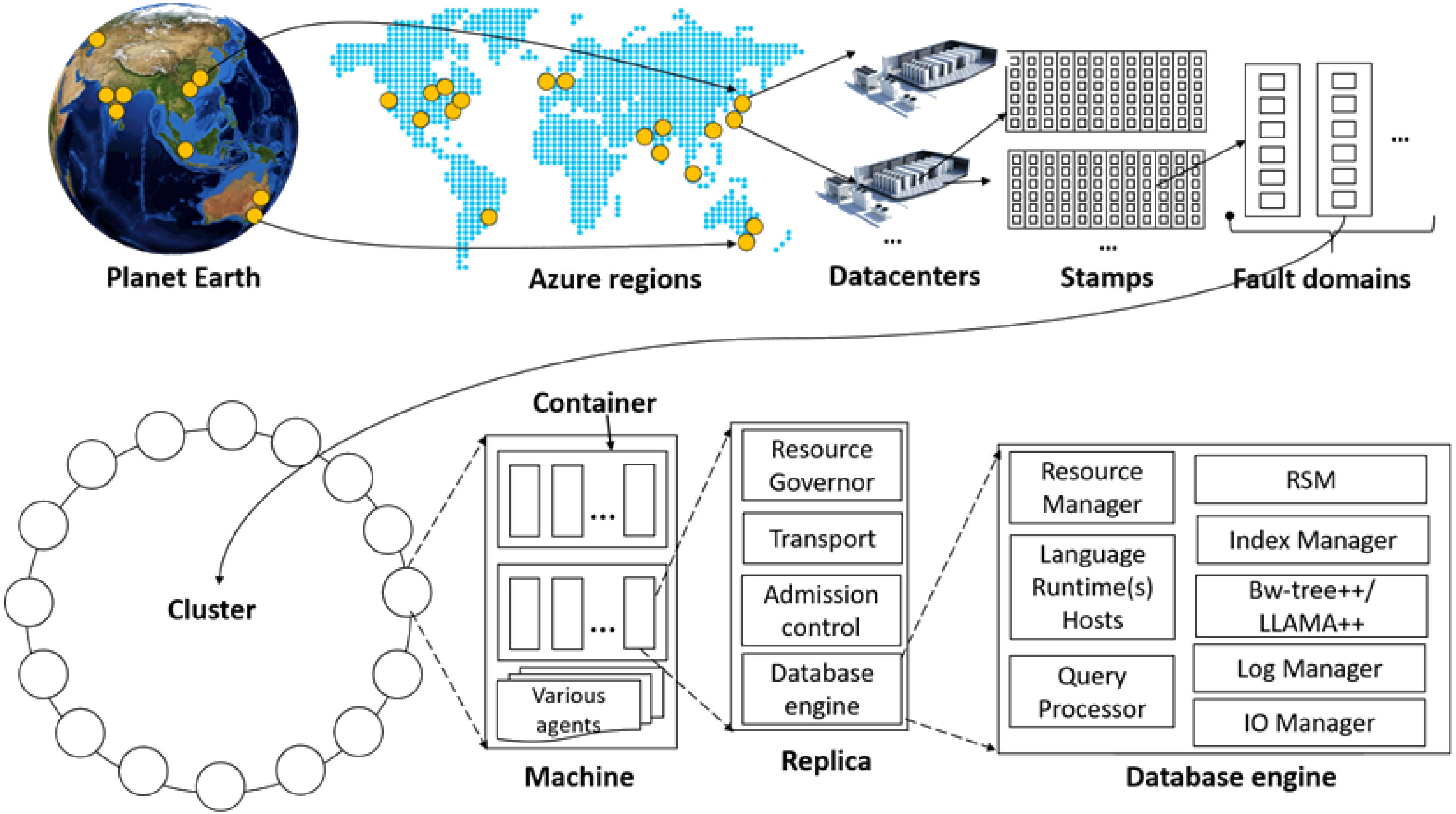
Elastic scale out of storage & throughput

Five well-defined consistency models

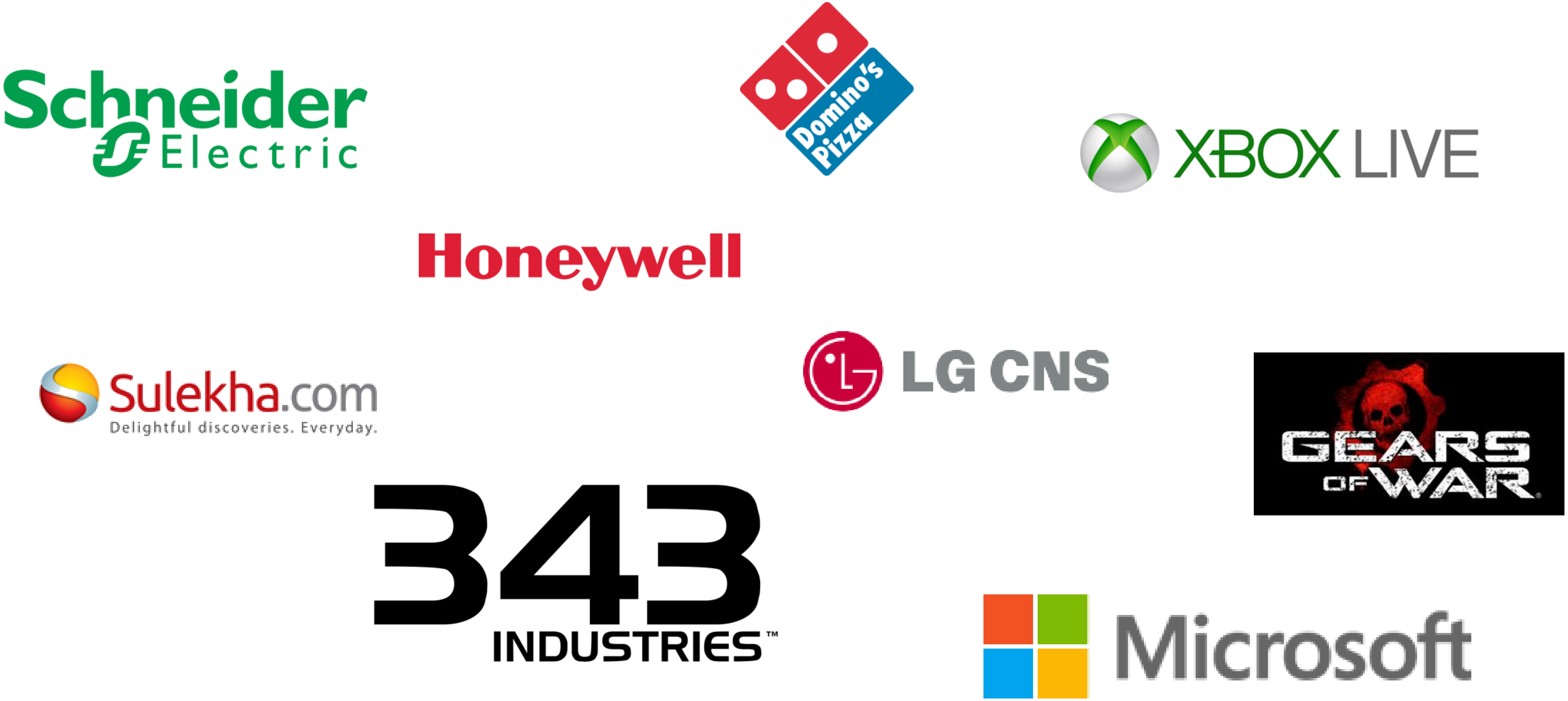
Turnkey global distribution

Comprehensive SLAs

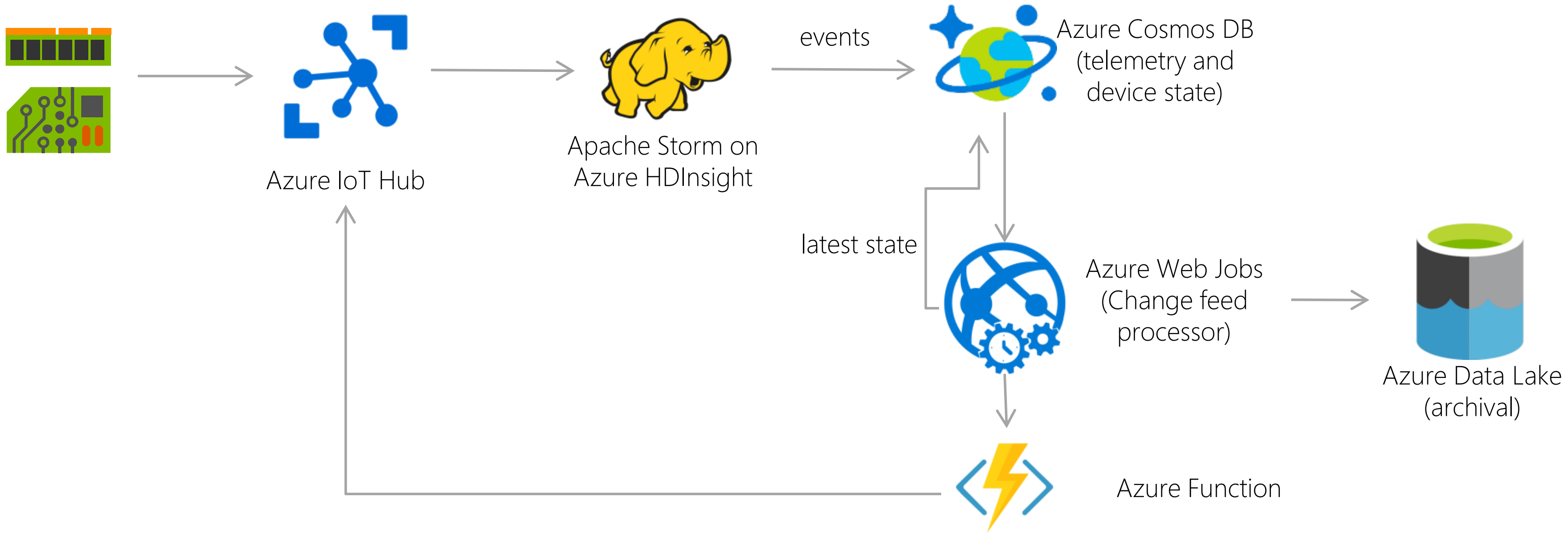
System topology (behind the scenes)



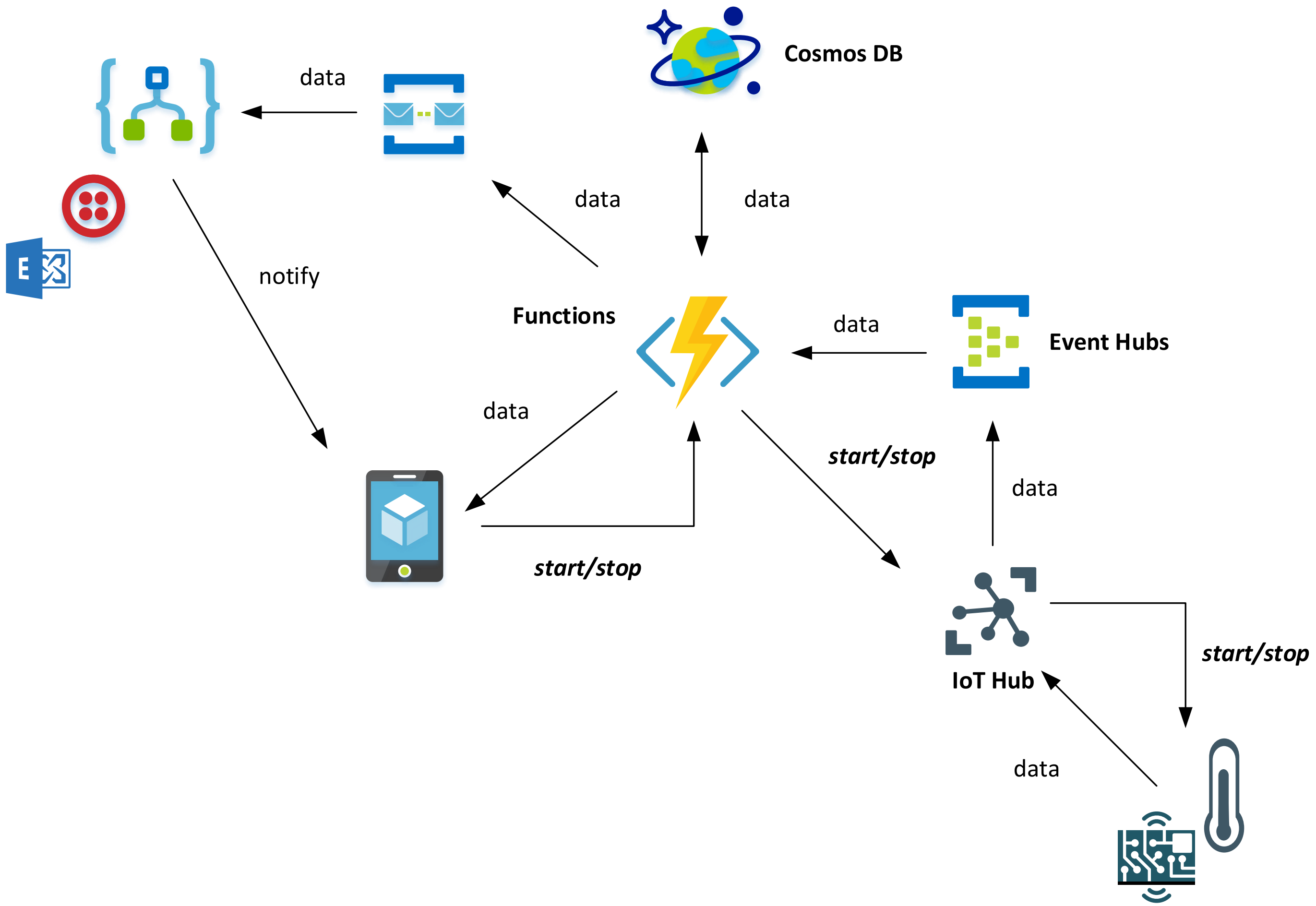
Customers


Schneider Electric
Domino's Pizza
XBOX LIVE
Honeywell
Sulekha.com
Delightful discoveries. Everyday.
LG CNS
GEARS OF WAR
343 INDUSTRIES™
Microsoft

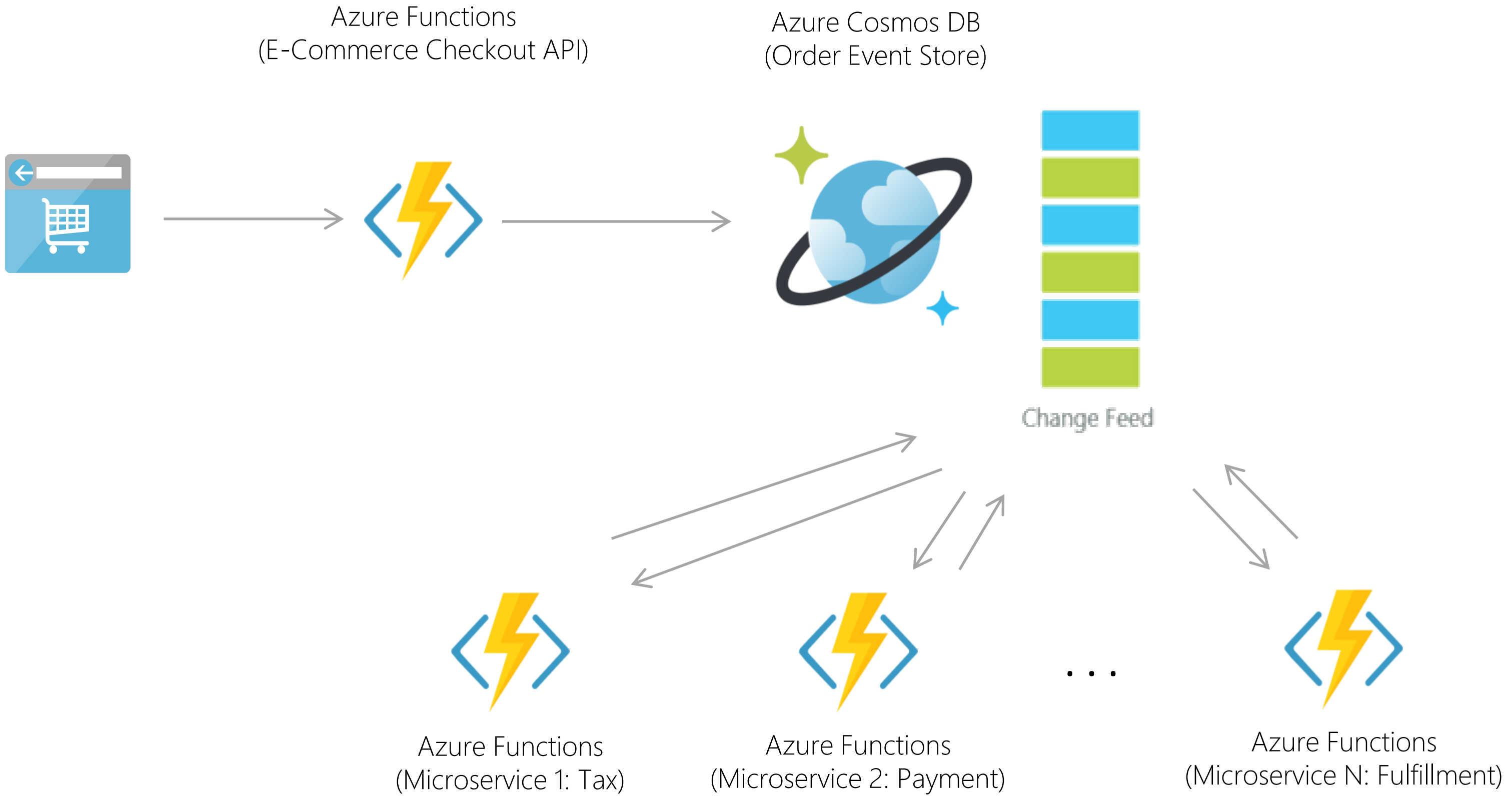
Scenario - Telemetry & Sensor Data


Azure IoT Hub
Apache Storm on Azure HDInsight
events
Azure Cosmos DB (telemetry and device state)
latest state
Azure Web Jobs (Change feed processor)
Azure Data Lake (archival)
Azure Function

Demo – IoT


Cosmos DB
data
data
data
notify
Functions
data
Event Hubs
data
start/stop
data
start/stop
IoT Hub
start/stop
data

Scenario – Order processing



Azure Functions
(E-Commerce Checkout API)

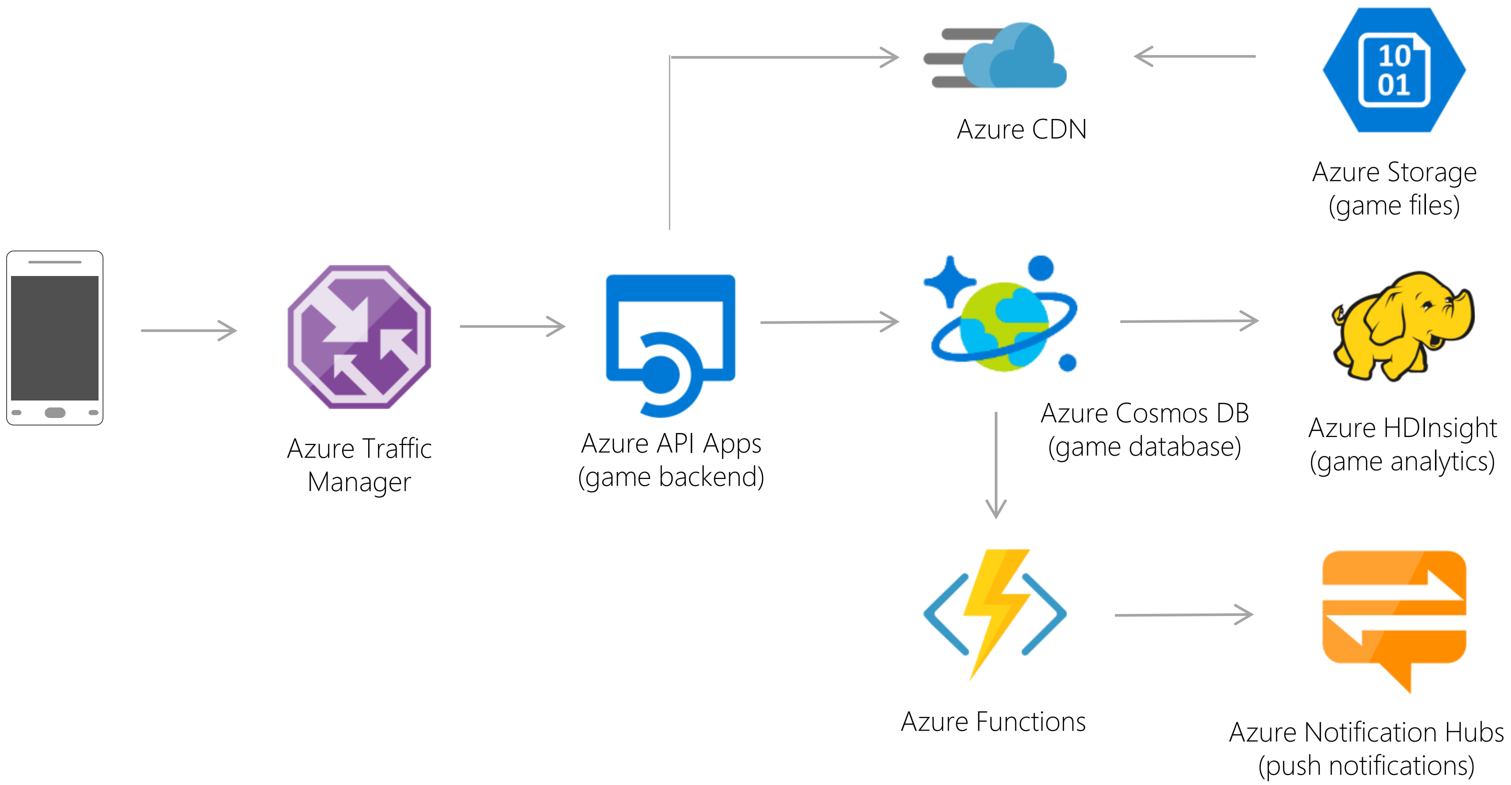
Azure Cosmos DB
(Order Event Store)

Azure Functions
(Microservice 1: Tax)

Azure Functions
(Microservice 2: Payment)

Azure Functions
(Microservice N: Fulfillment)

Scenario – Multiplayer gaming



Cosmos DB Characteristics

Global distribution

Resource model

Scale

Consistency

Indexing

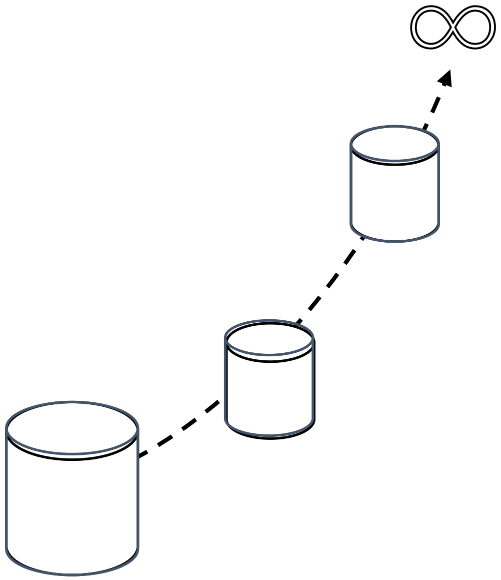
Global Distribution

High Availability

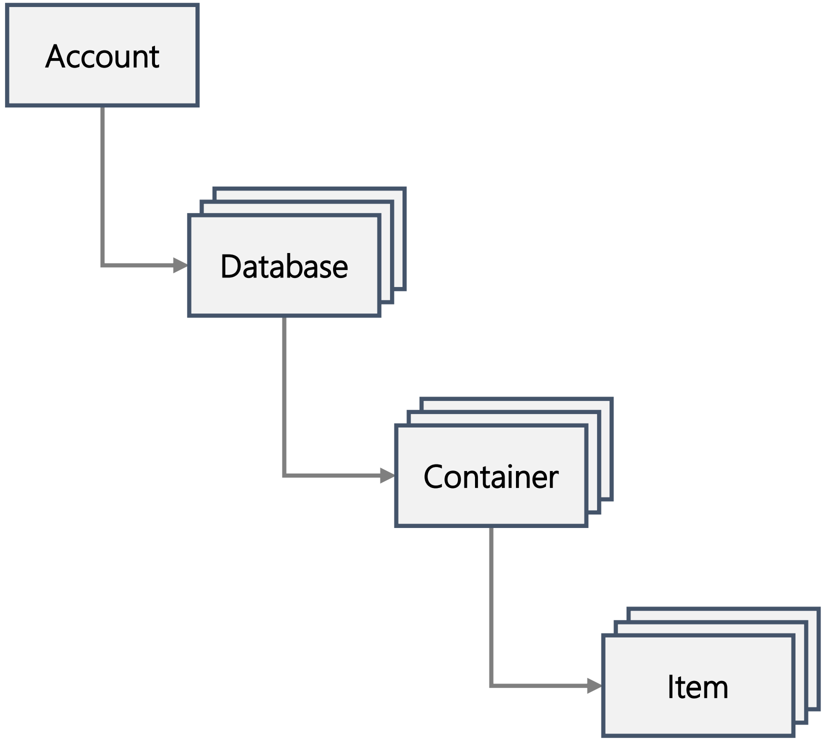
- Automatic and Manual Failover
- Multi-homing API removes need for app redeployment

Low Latency (anywhere in the world)

- Sending a packet across the world under ideal network conditions takes 100's of milliseconds
- Packets cannot move fast than the speed of light

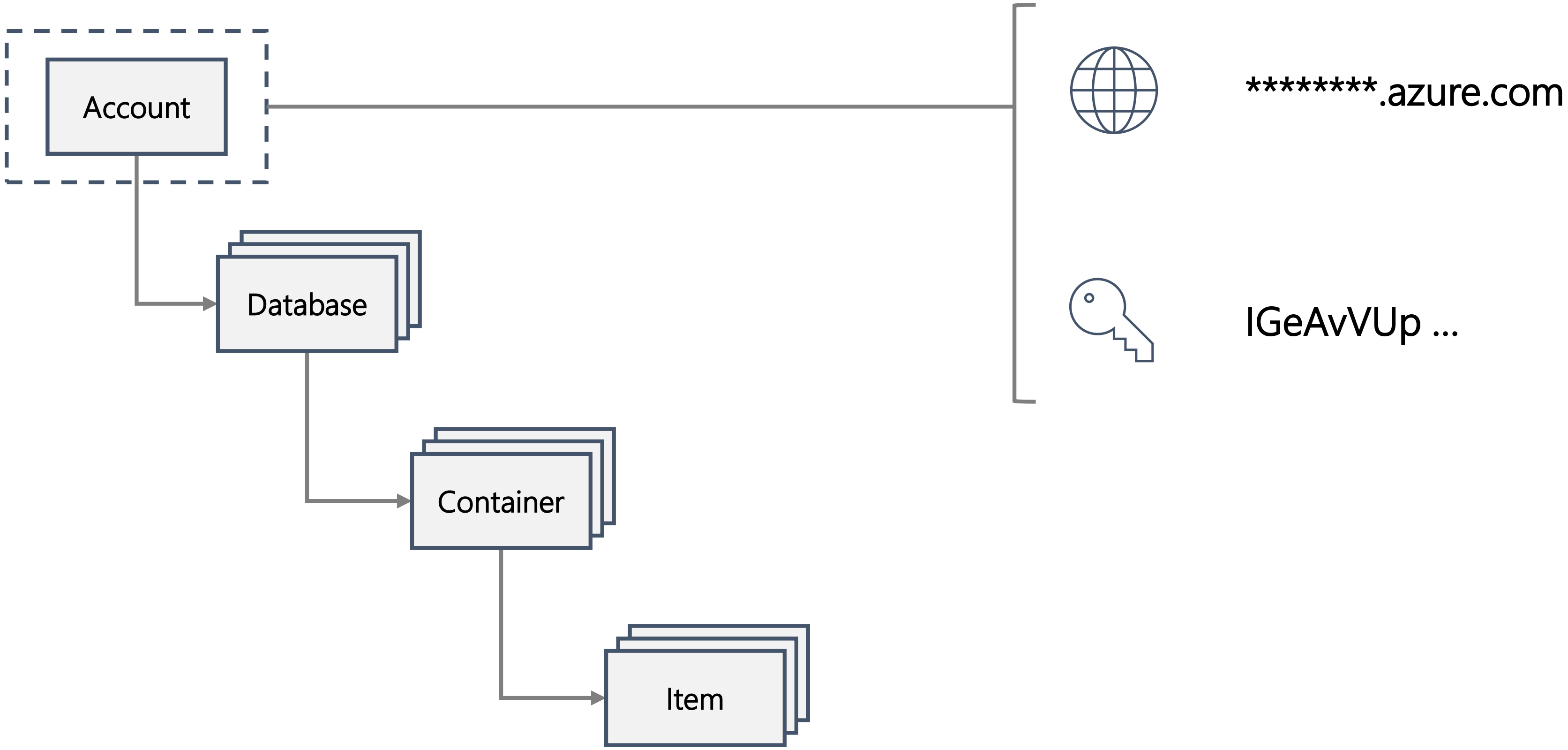


Resource Model

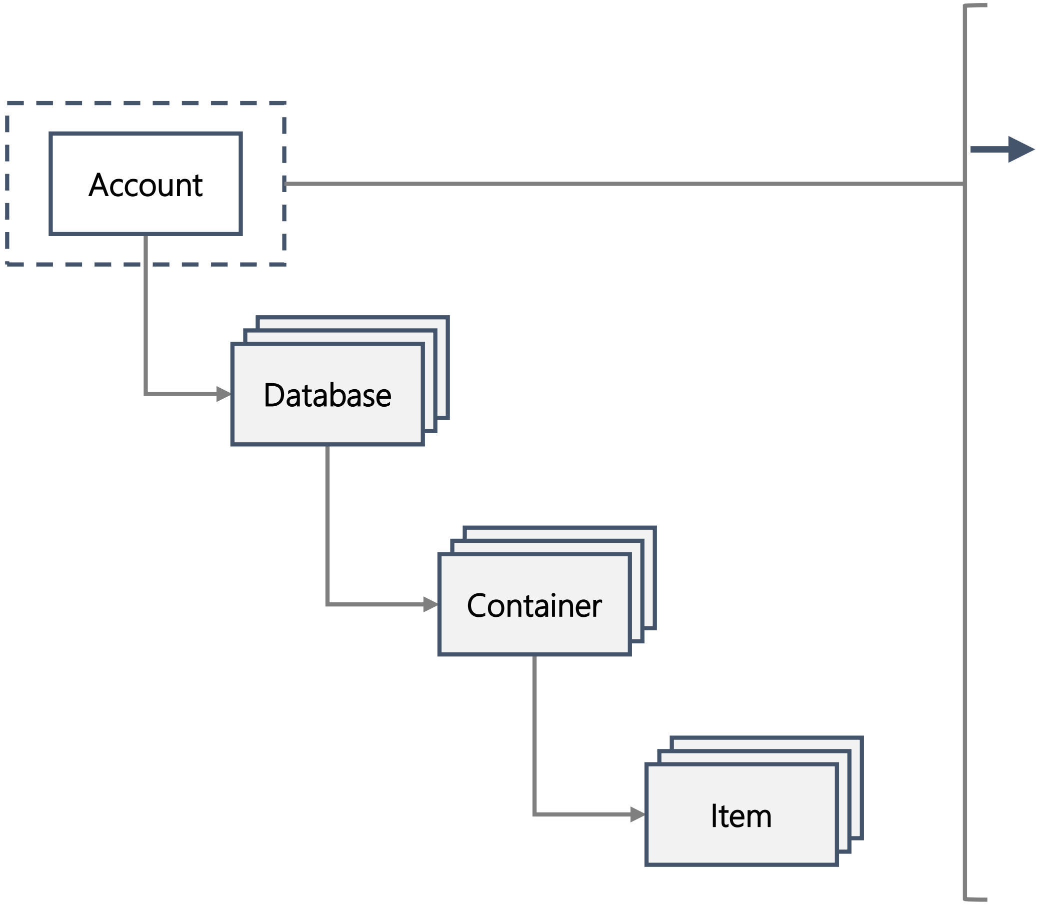


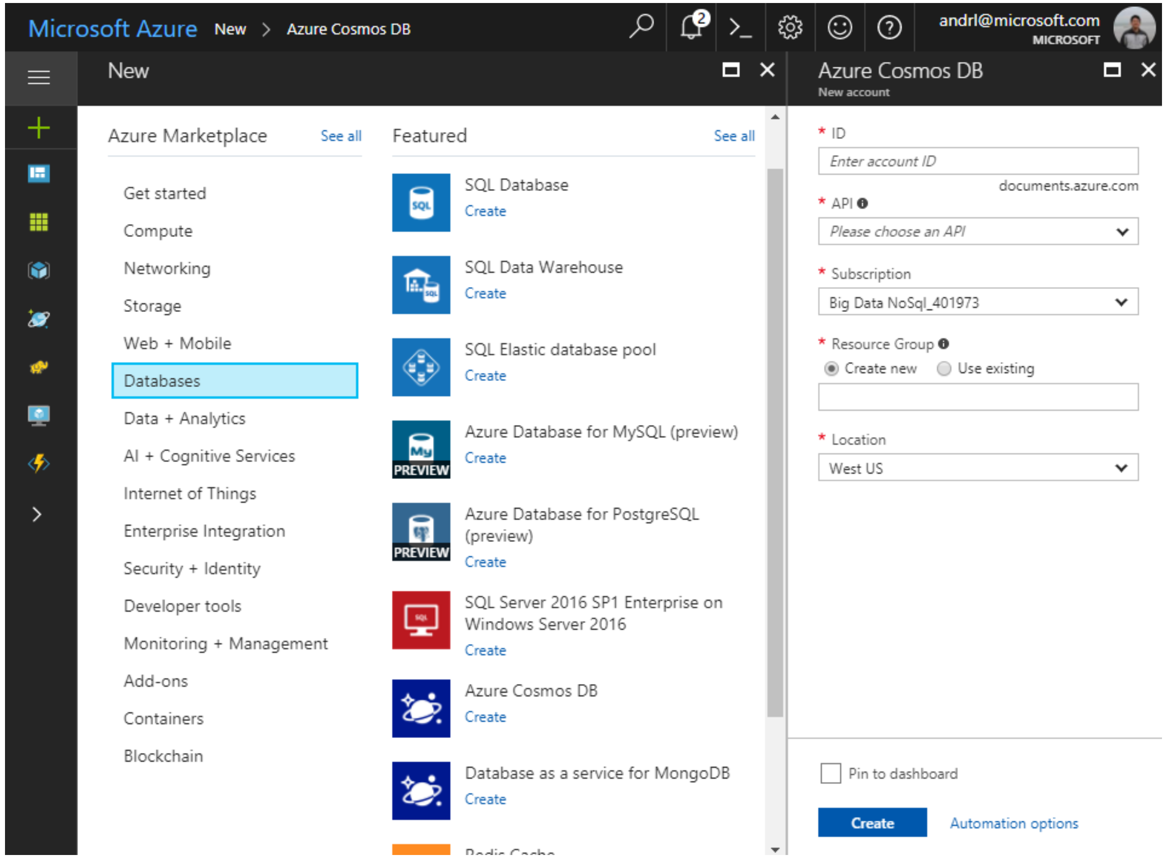
- Resources identified by their logical and stable URI
- Hierarchical overlay over horizontally partitioned entities; spanning machines, clusters and regions
- Extensible custom projections based on specific type of API interface
- Stateless interaction (HTTP and TCP)

Account URI and Credentials



Creating Account





* ID

Enter account ID

documents.azure.com

* API

Please choose an API

* Subscription

Big Data NoSql_401973

* Resource Group

● Create new ○ Use existing

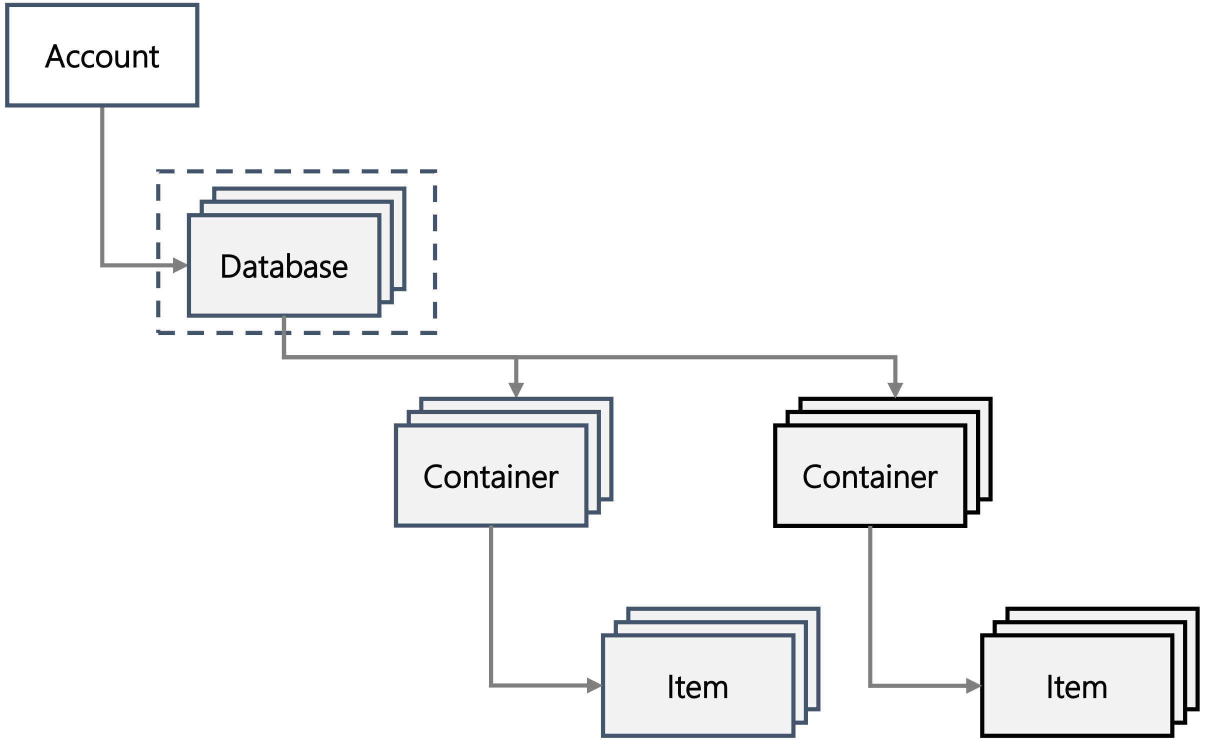
* Location

West US

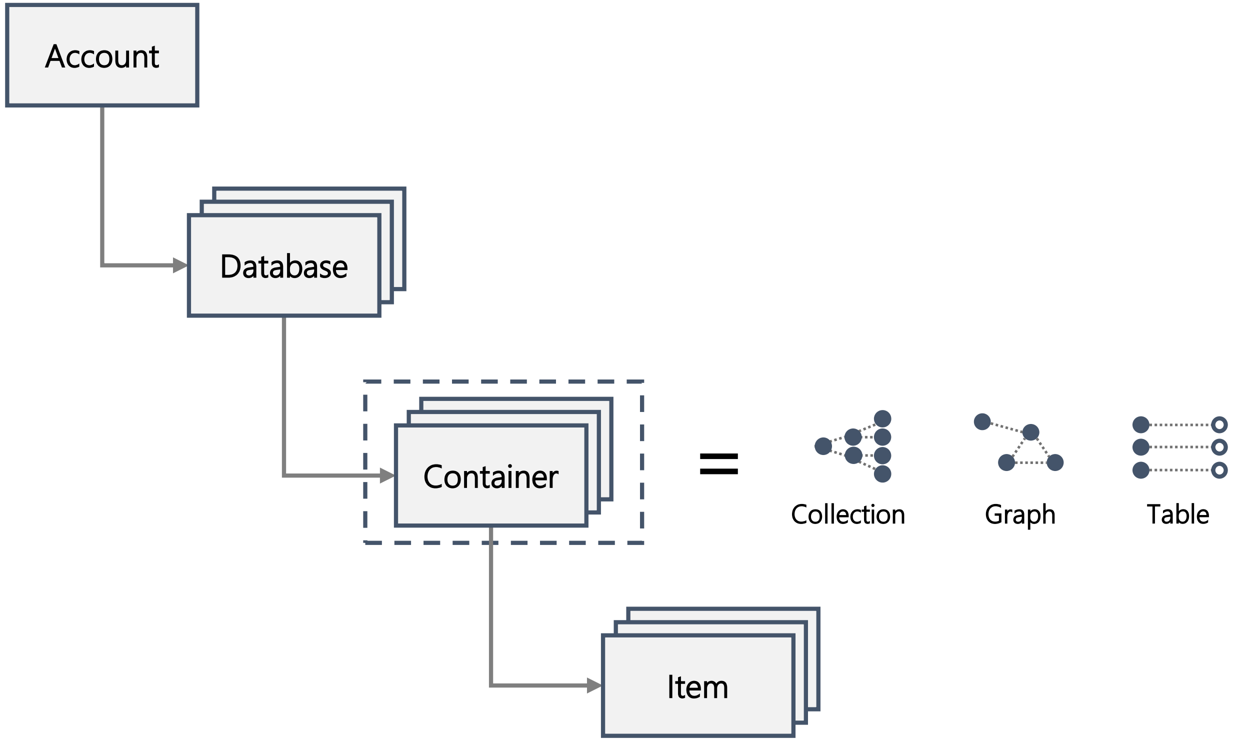
□ Pin to dashboard

Create   Automation options

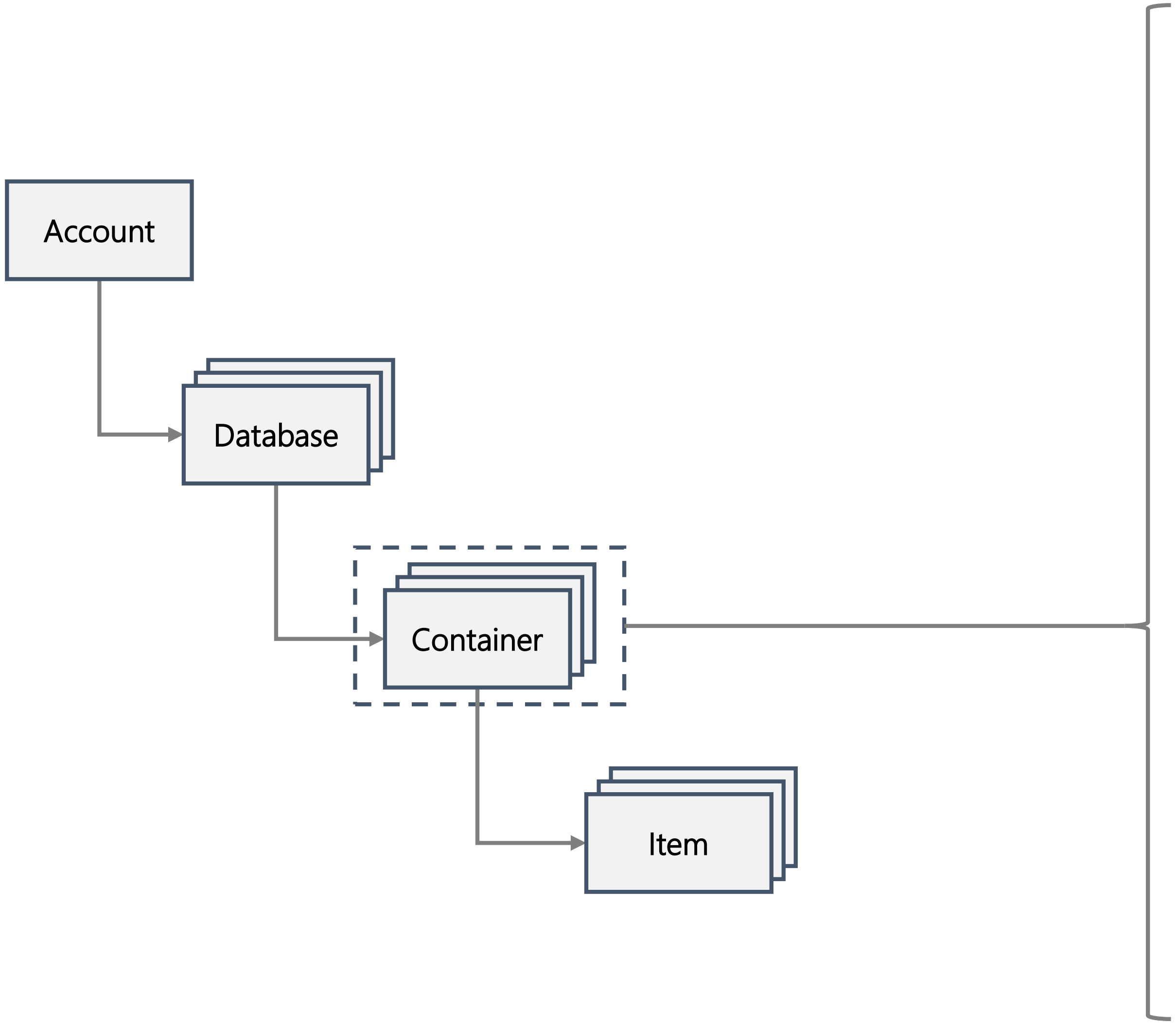
Database representation

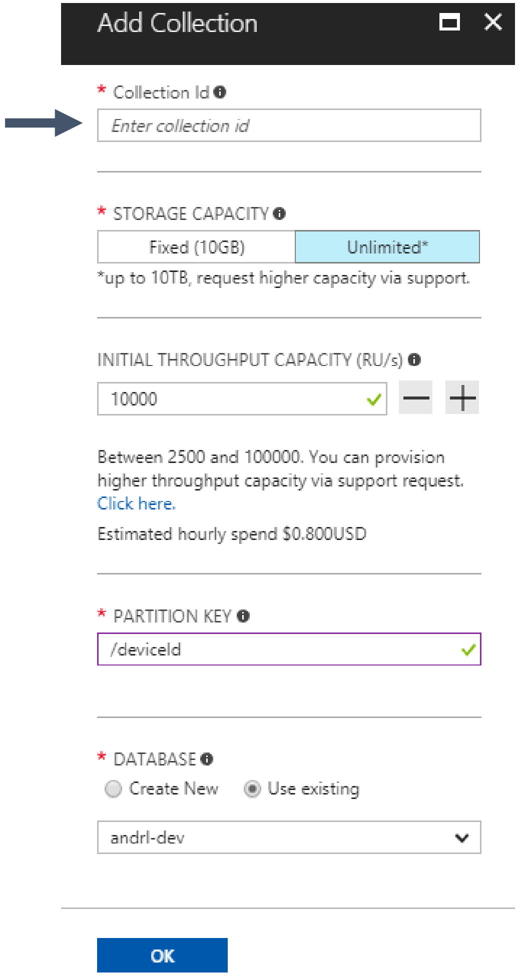


Container representation



Creating collection – SQL API





* Collection Id

Enter collection id

* STORAGE CAPACITY

Fixed (10GB) | [✓ Unlimited*]

*up to 10TB, request higher capacity via support.

INITIAL THROUGHPUT CAPACITY (RU/s)

10000 ✓

Between 2500 and 100000. You can provision higher throughput capacity via support request. Click here.

Estimated hourly spend $0.800USD

* PARTITION KEY

/deviceId ✓

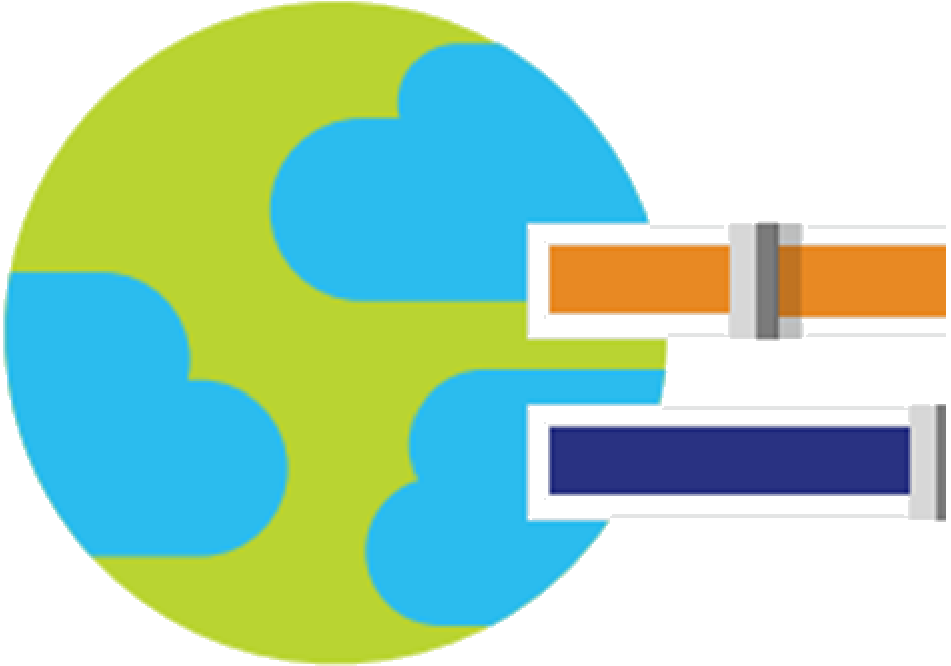
* DATABASE

○ Create New ● Use existing

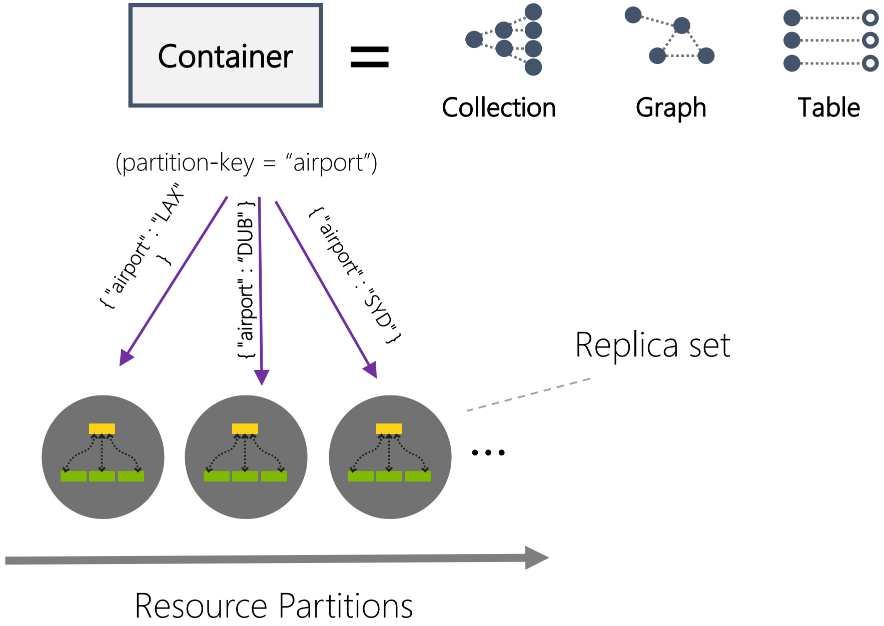
andrl-dev

Scale

- Pay as go for storage and throughput
- Elastic scale across regions
- Partitions



Horizontal Scaling

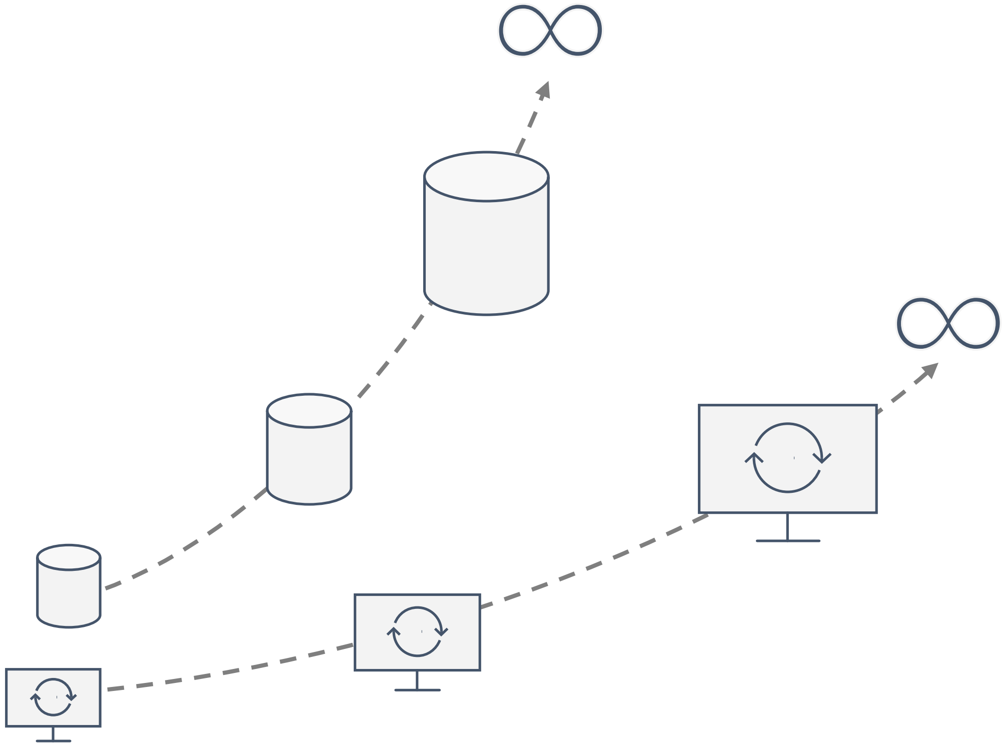


- Containers are horizontally partitioned
- Each partition made highly available via a replica set
- Partition management is transparent and highly responsive
- Partitioning scheme is dictated by a "partition-key"

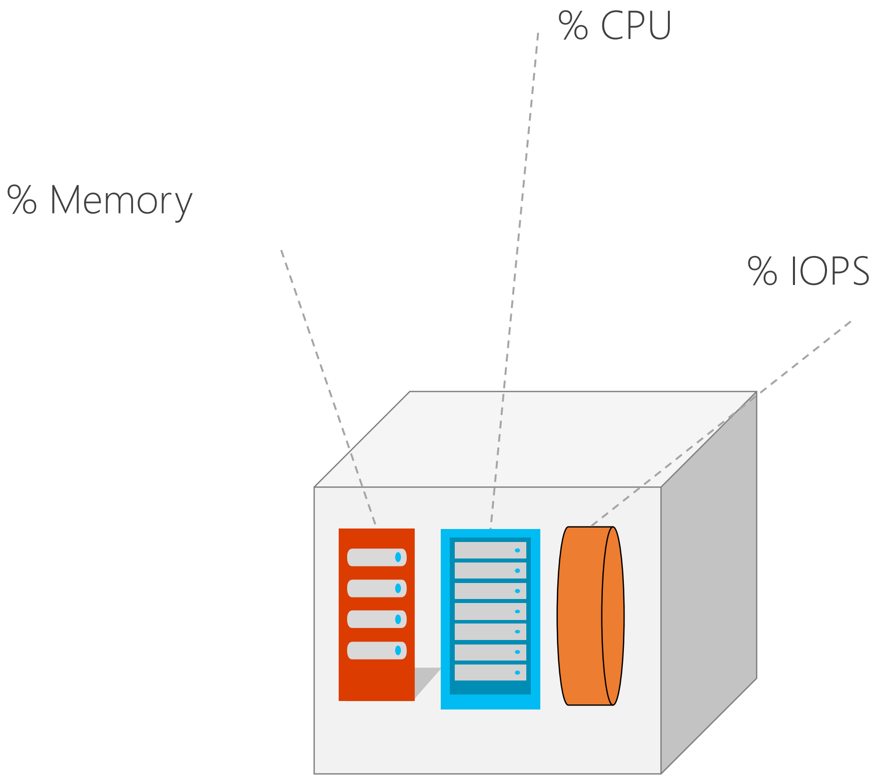
Elastic Scale Out of Storage and Throughput

SCALES AS YOUR APPS' NEEDS CHANGE

- Database elastically scales storage and throughput
- How? Scale-out!
- Collections can span across large clusters of machines
- Can start small and seamlessly grow as your app grows

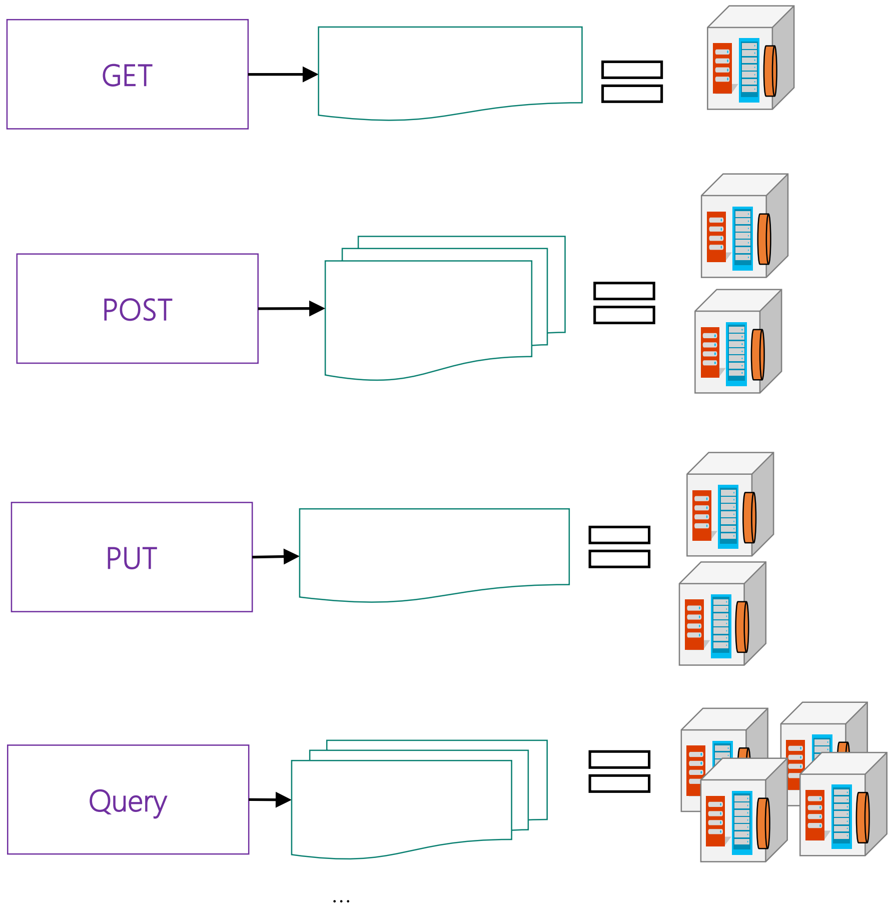


Request Units



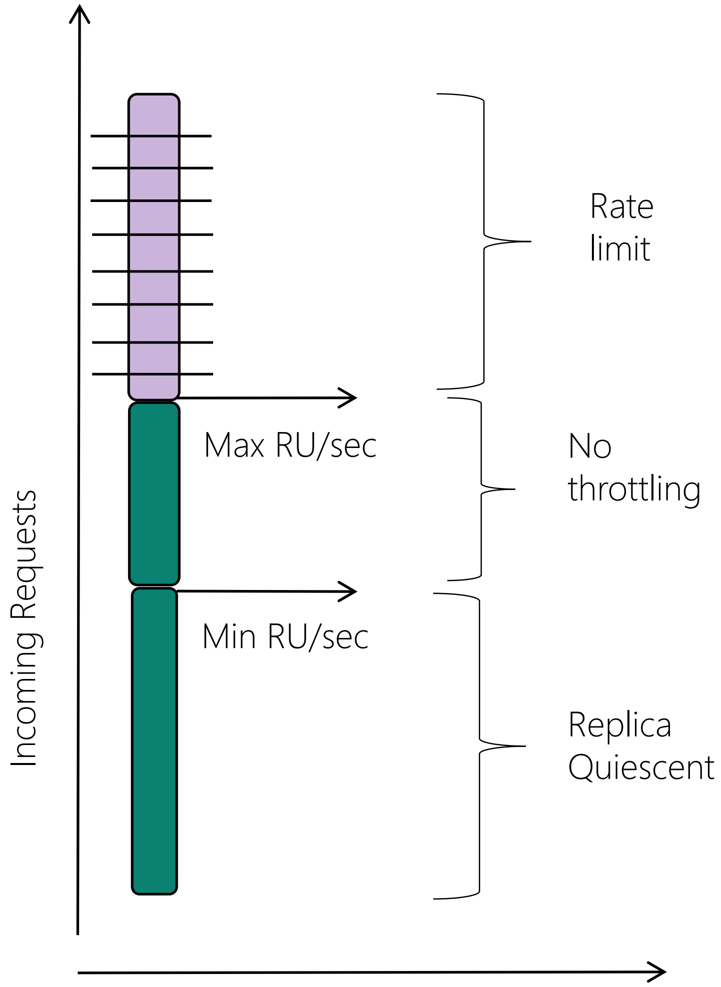
- Request Units (RU) is a rate-based currency
- Abstracts physical resources for performing requests
- Key to multi-tenancy, SLAs, and COGS efficiency
- Foreground and background activities

Request Units



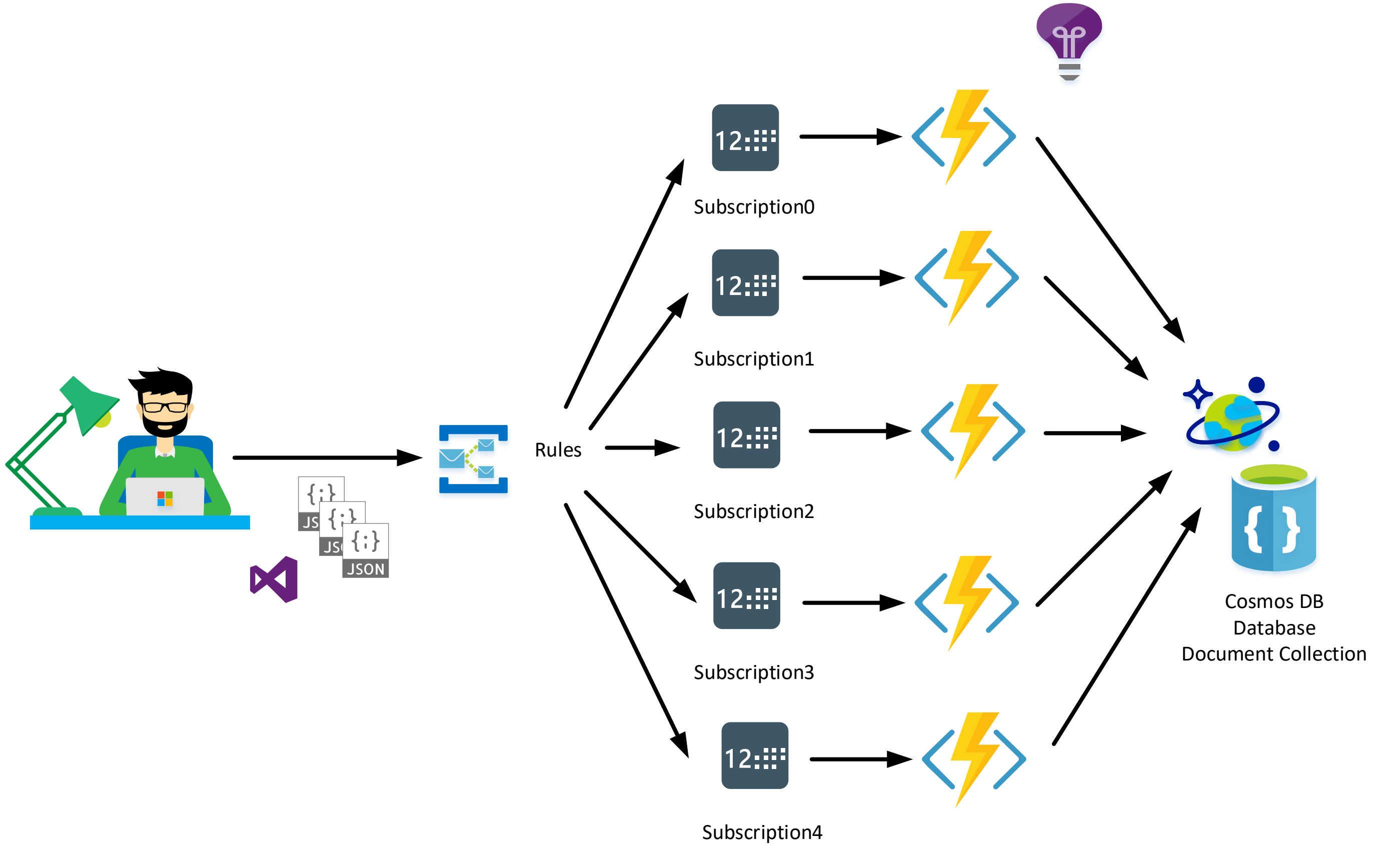
- Normalized across various access methods
- 1 RU = 1 read of 1 KB document
- Each request consumes fixed RUs
- Applies to reads, writes, queries, and stored procedure execution

Request Units

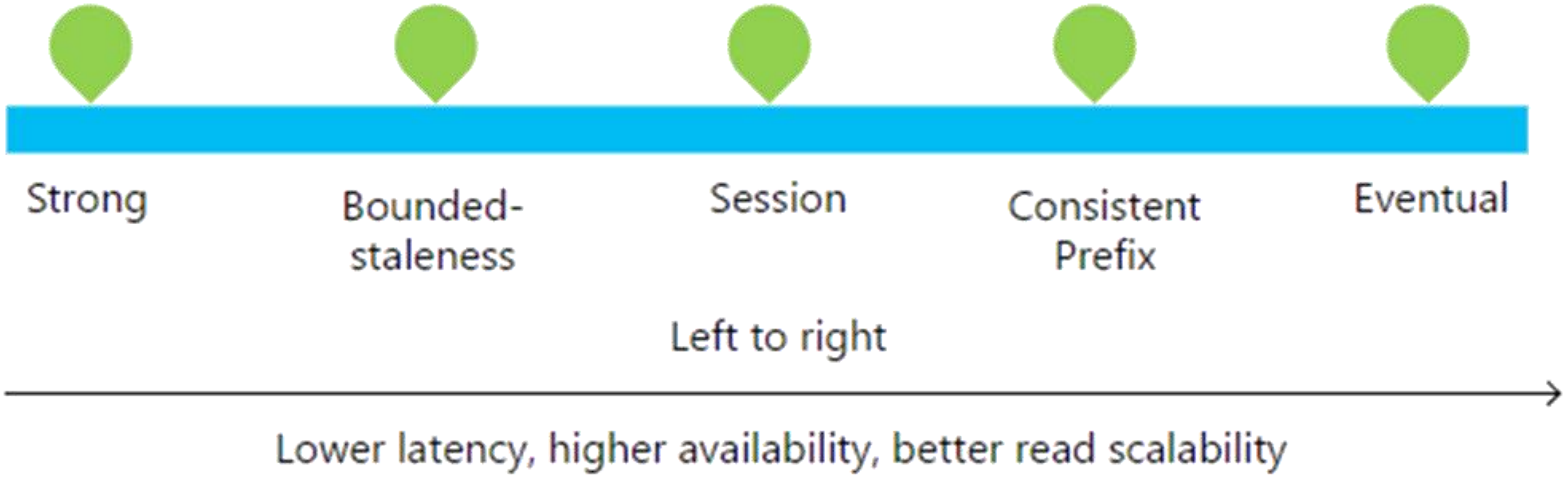


- Provisioned in terms of RU/sec and RU/min granularities
- Rate limiting based on amount of throughput provisioned
- Can be increased or decreased instantaneously
- Metered Hourly
- Background processes like TTL expiration, index transformations scheduled when quiescent

Demo - Scale


JSON
Rules
12
Subscription0
Subscription1
Subscription2
Subscription3
Subscription4
Cosmos DB
Database
Document Collection

5 Well-defined, consistency models

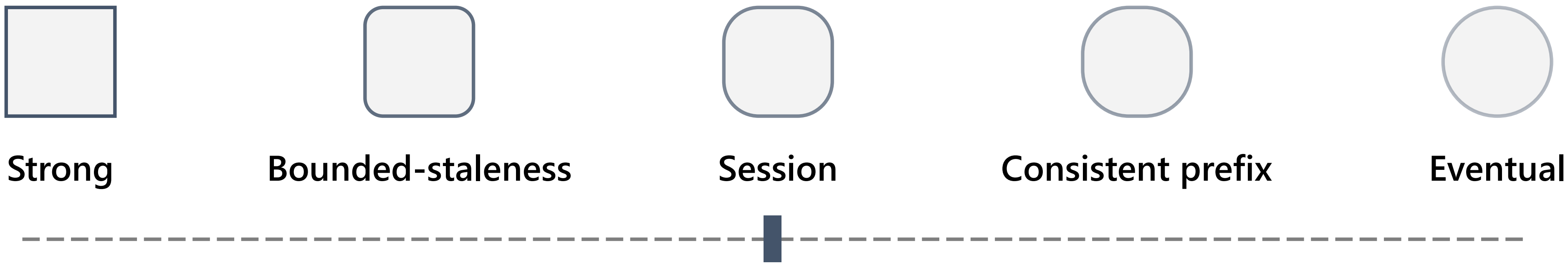


Choose the right consistency

- Overridable on a per-request basis
- Provides control over performance-consistency tradeoffs, backed by comprehensive SLAs.
- An intuitive programming model offering low latency and high availability for your planet-scale app.

CLEAR TRADEOFFS

- Latency
- Availability
- Throughput



Strong

Bounded-staleness

Session

Consistent prefix

Eventual

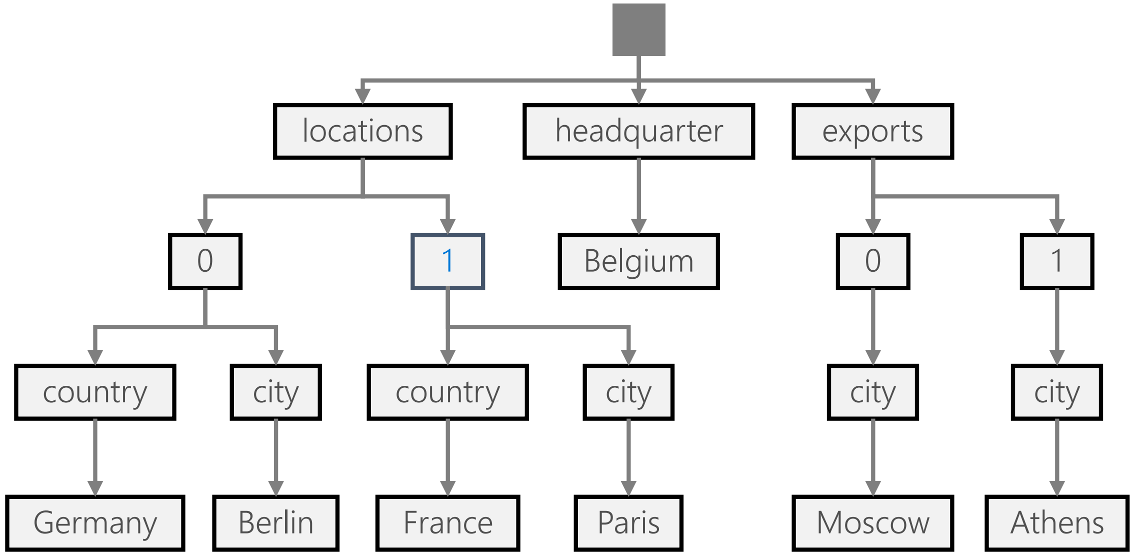
Handle any data with no schema or indexing required

Azure Cosmos DB’s schema-less service automatically indexes all your data, regardless of the data model, to delivery fast queries.

- Automatic index management
- Synchronous auto-indexing
- No schemas or secondary indices needed
- Works across every data model

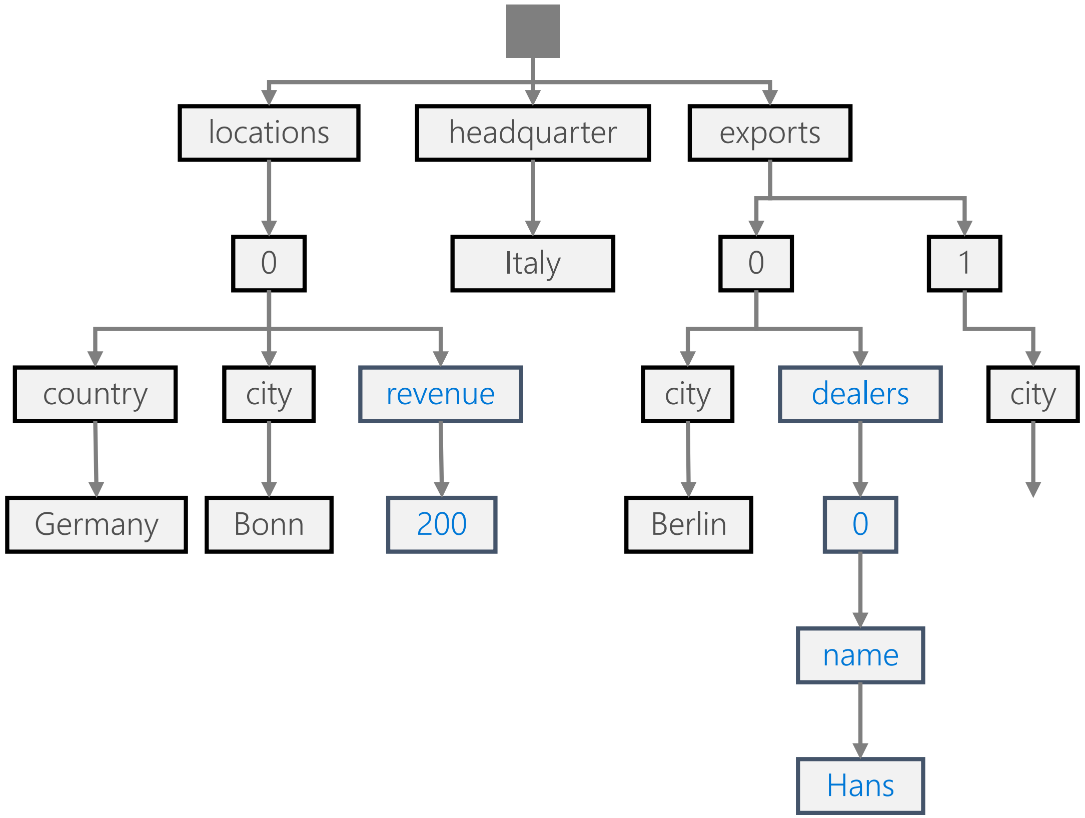
Indexing Json Documents

```
{
    "locations": [
        {
            "country": "Germany",
            "city": "Berlin"
        },
        {
            "country": "France",
            "city": "Paris"
        }
    ],
    "headquarter": "Belgium",
    "exports": [
        { "city": "Moscow" },
        { "city": "Athens" }
     ]
}
```

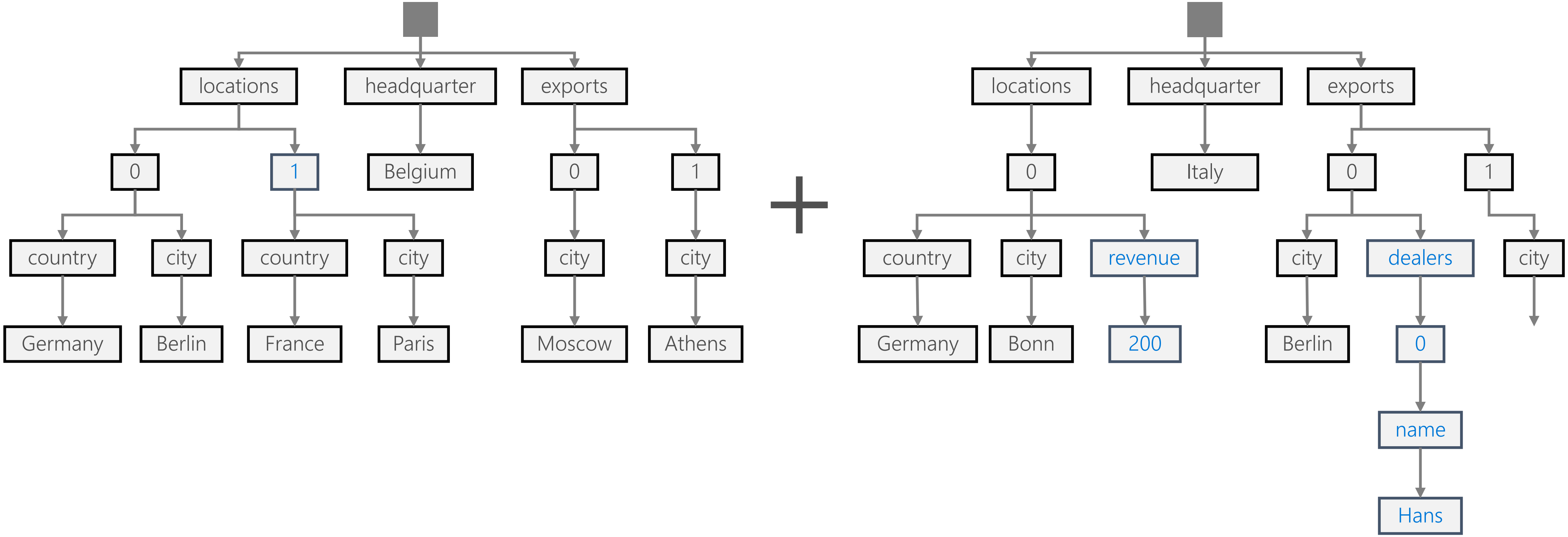


Indexing Json Documents - continued

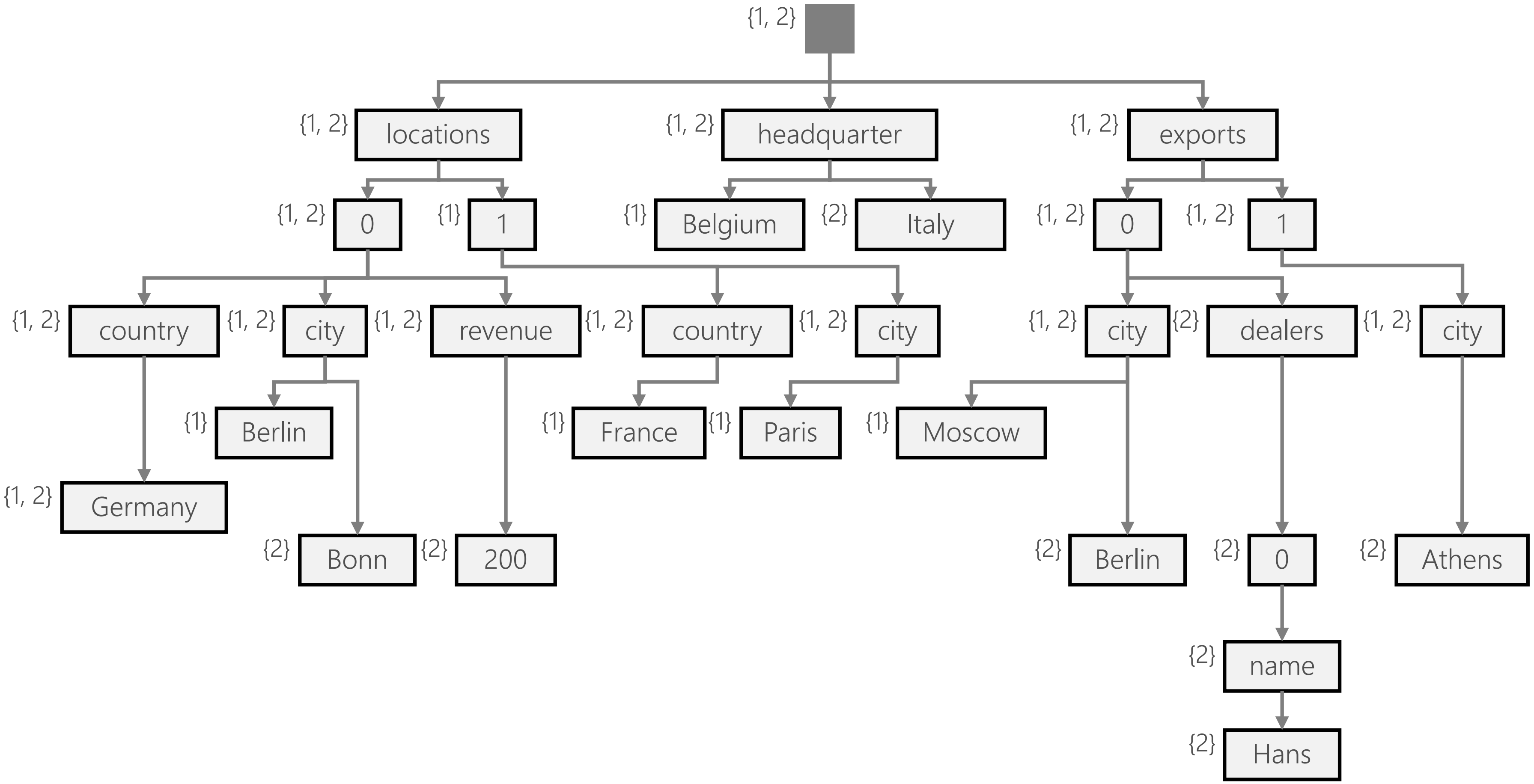
```
{
    "locations": [
        {
            "country": "Germany",
            "city": "Bonn",
            "revenue": 200
        }
    ],
    "headquarter": "Italy",
    "exports": [
        {
            "city": "Berlin",
            "dealers": [
                { "name": "Hans" }
            ]
        },
        { "city": "Athens" }
    ]
}
```



Indexing Json Documents - continued



Inverted Index



Indexing policies

CUSTOM INDEXING POLICIES

Though all Azure Cosmos DB data is indexed by default, you can specify a custom indexing policy for your collections.

Custom indexing policies allow you to design and customize the shape of your index while maintaining schema flexibility.

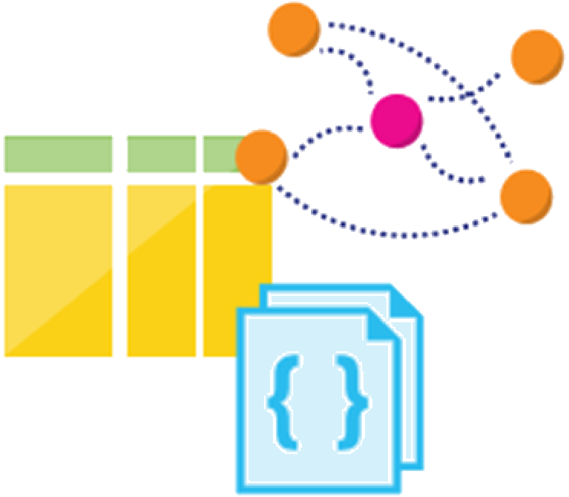
- Define trade-offs between storage, write and query performance, and query consistency
- Include or exclude documents and paths to and from the index
- Configure various index types

```
{
    "automatic": true,
    "indexingMode": "Consistent",
    "includedPaths": [{
        "path": "/*",
        "indexes": [{
            "kind": "Hash",
            "dataType": "String",
            "precision": -1
        }, {
            "kind": "Range",
            "dataType": "Number",
            "precision": -1
        }, {
            "kind": "Spatial",
            "dataType": "Point"
        }]
    }],
    "excludedPaths": [{
        "path": "/nonIndexedContent/*"
    }]
}
```

Swiss Army Knife

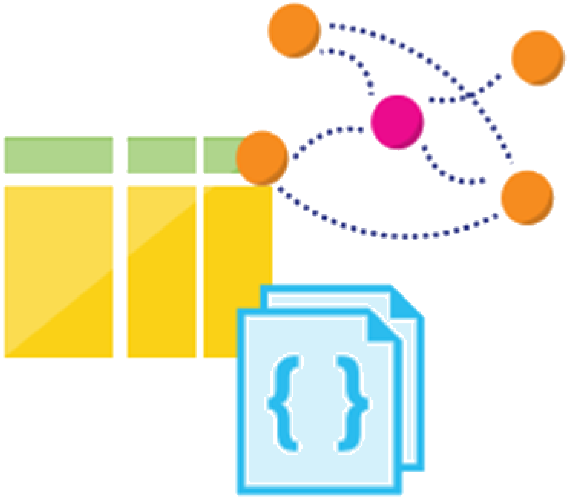
Multi-model + Multi API

Reference Case



Multi-model + multi-API

- Different models:
  - Graph
  - Key-Value
  - Document DB

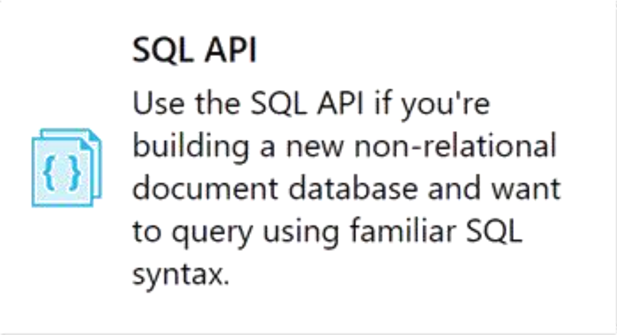


- API support:
  - SQL
  - JavaScript
  - Gremlin
  - MongoDB
  - Azure Table Storage
  - Cassandra

Mimicking Strategy

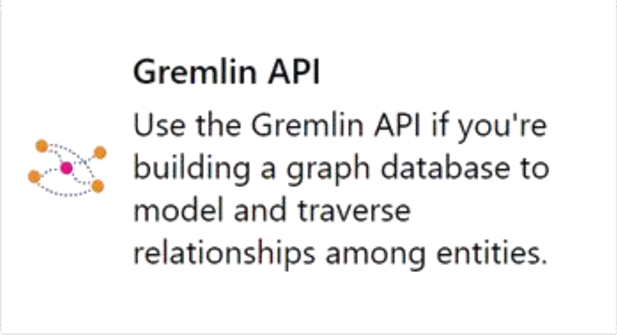
- MongoDB
- Cassandra

What model?



SQL API

Use the SQL API if you're building a new non-relational document database and want to query using familiar SQL syntax.



Gremlin API

Use the Gremlin API if you're building a graph database to model and traverse relationships among entities.

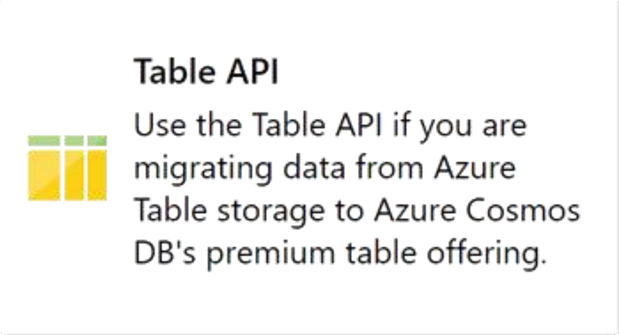
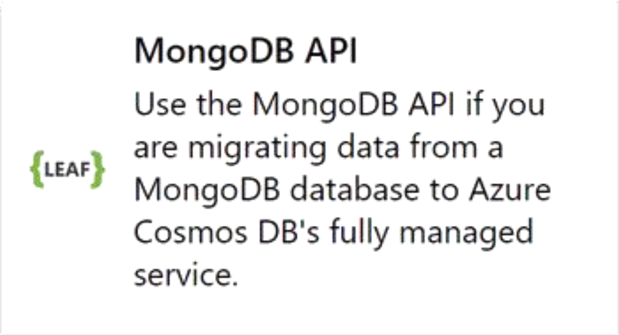


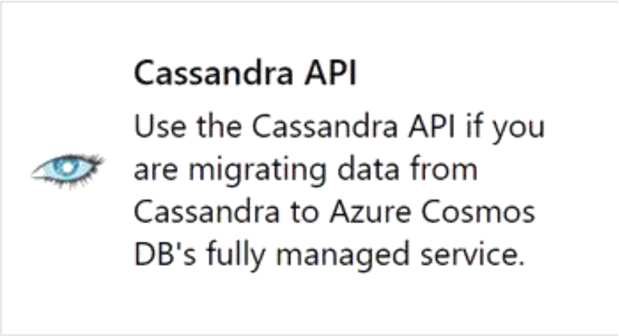
Table API

Use the Table API if you are migrating data from Azure Table storage to Azure Cosmos DB's premium table offering.



MongoDB API

Use the MongoDB API if you are migrating data from a MongoDB database to Azure Cosmos DB's fully managed service.



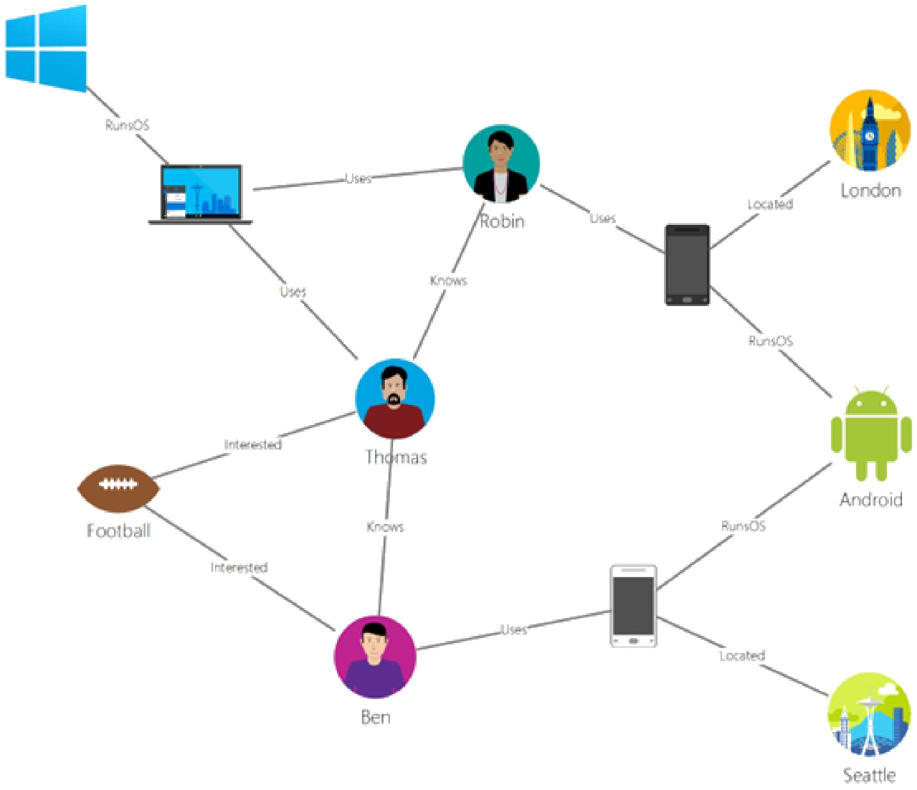
Cassandra API

Use the Cassandra API if you are migrating data from Cassandra to Azure Cosmos DB's fully managed service.

Document (JSON)

- A schema-less JSON database engine with rich SQL querying capabilities.
- Store documents
- Searchable by integrating with Azure Search
- Easy integration with Azure Functions
- Change Feed

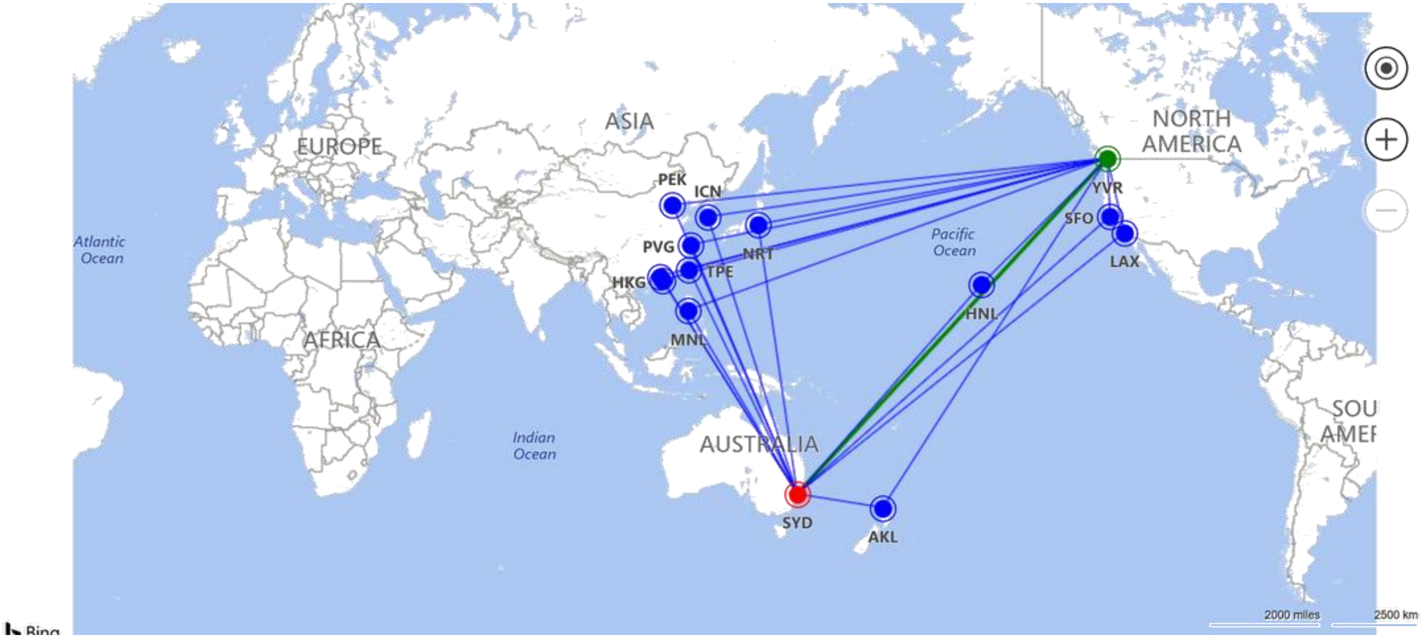
Graph model



A **graph** is a structure that's composed of vertices and edges. Both ***vertices*** and ***edges*** can have an arbitrary number of properties.

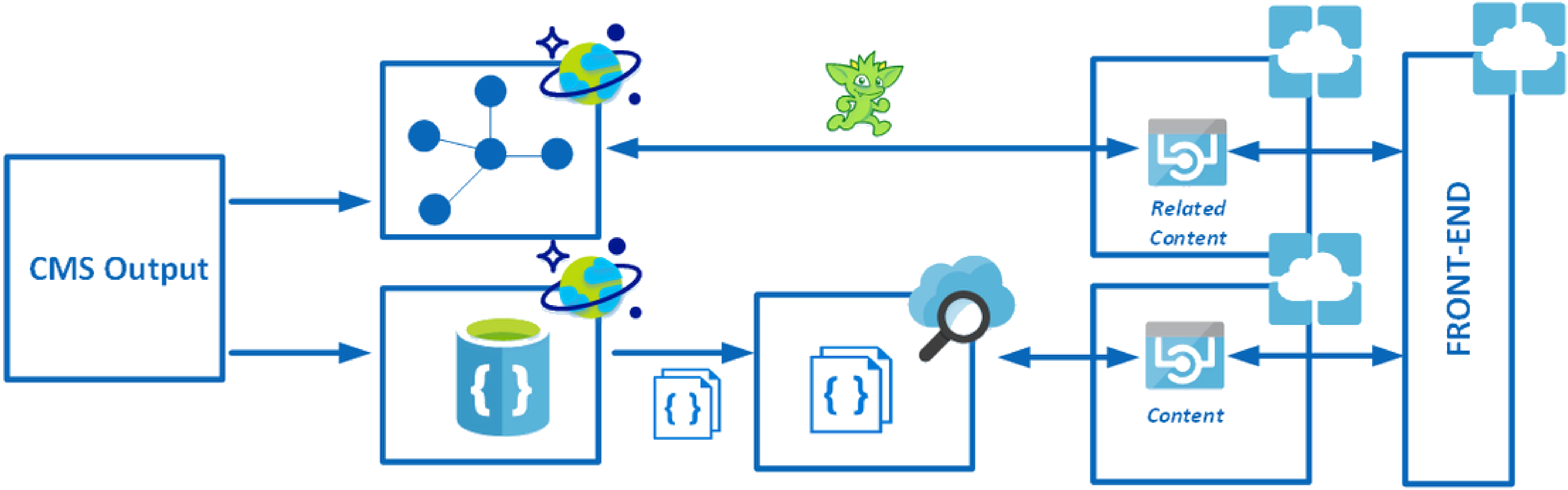
- **Vertices** - Vertices denote discrete objects, such as a person, a place, or an event.
- **Edges** - Edges denote relationships between vertices. For example, a person might know another person, be involved in an event, and recently been at a location.
- **Properties** - Properties express information about the vertices and edges.

CHI
CA
GO
Chicago Marathon
J 58069
10·7·18


EUROPE
ASIA
NORTH AMERICA
AFRICA
AUSTRALIA
SOU
AMEF
Atlantic Ocean
Indian Ocean
Pacific Ocean
PEK
ICN
NRT
PVG
TPE
HKG
MNL
YVR
SFO
LAX
HNL
SYD
AKL
Bing
2000 miles
2500 km

Demo - Graph

Use case – Cosmos DB Graph



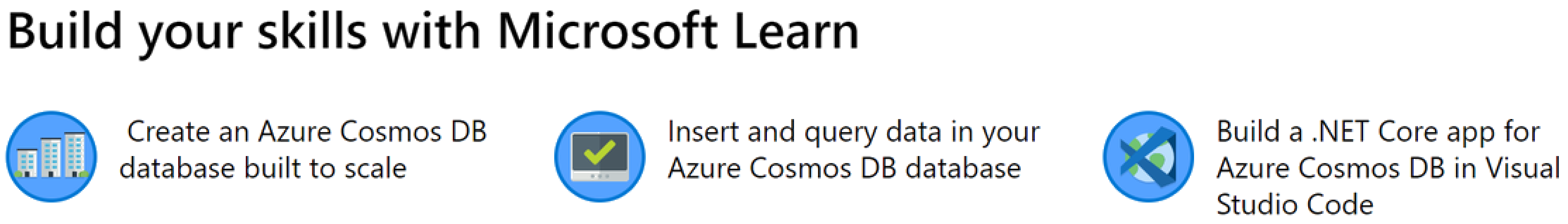
https://customers.microsoft.com/en-in/story/reed-business-information-professional-services-azure

Key Takeaways

- Multiple options with models and API’s
- Various consistency models
- Global scale
- Flexible through put
- Support for diverse scenario’s

Call to action

- Documentation: https://docs.microsoft.com/en-us/azure/
- Middleware Friday: https://www.youtube.com/watch?v=ZplZgOoGUzY
- Graph demo: https://github.com/anthonychu/cosmosdb-gremlin-flights
- Pluralsight: https://www.pluralsight.com/courses/azure-cosmos-db
- Channel 9: https://channel9.msdn.com/Events/Build/2018/BRK3319



Build your skills with Microsoft Learn

Create an Azure Cosmos DB database built to scale

Insert and query data in your Azure Cosmos DB database

Build a .NET Core app for Azure Cosmos DB in Visual Studio Code

Running on time

Azure
Lowlands

Thanks and have a great day!