

# Select Stars: A SQL DBA's Introduction to Azure Cosmos DB

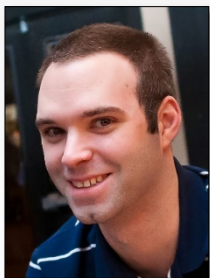
---

**sqlbits**

Bob Pusateri

**HERAFLUX**  
TECHNOLOGIES®

# About Bob Pusateri



**Microsoft**  
**CERTIFIED**

Master

SQL Server 2008



@sqlbob



bob@bobpusateri.com



heraflux.com



bobpusateri



## Specialties / Focus Areas / Passions:

- Performance Tuning & Troubleshooting
- Very Large Databases
- SQL Server Storage Engine
- HA/DR
- Cloud





# FEEDBACK FORMS

PLEASE FILL OUT AND PASS TO YOUR ROOM  
HELPER BEFORE YOU LEAVE THE SESSION

# What if...

---

**We were developing an IoT system**

...Which needed to ingest data from thousands/millions of devices

... and that data needed to be queried within seconds?



# What if...

---

**We were building an e-commerce site**

Which needed guaranteed performance and availability

... anywhere on Earth

... and needed to be able to scale up/down in response to conditions?



# How Would We Manage The Data?

---



# Agenda

---

**What is  
Azure Cosmos DB?**

**What's It Offer?**

**How Does It Work?**

**Partitioning**

**Consistency**

**Indexing**

**Backups**

**What's It Cost?**

# What is Azure Cosmos DB?



# What Is Azure Cosmos DB?

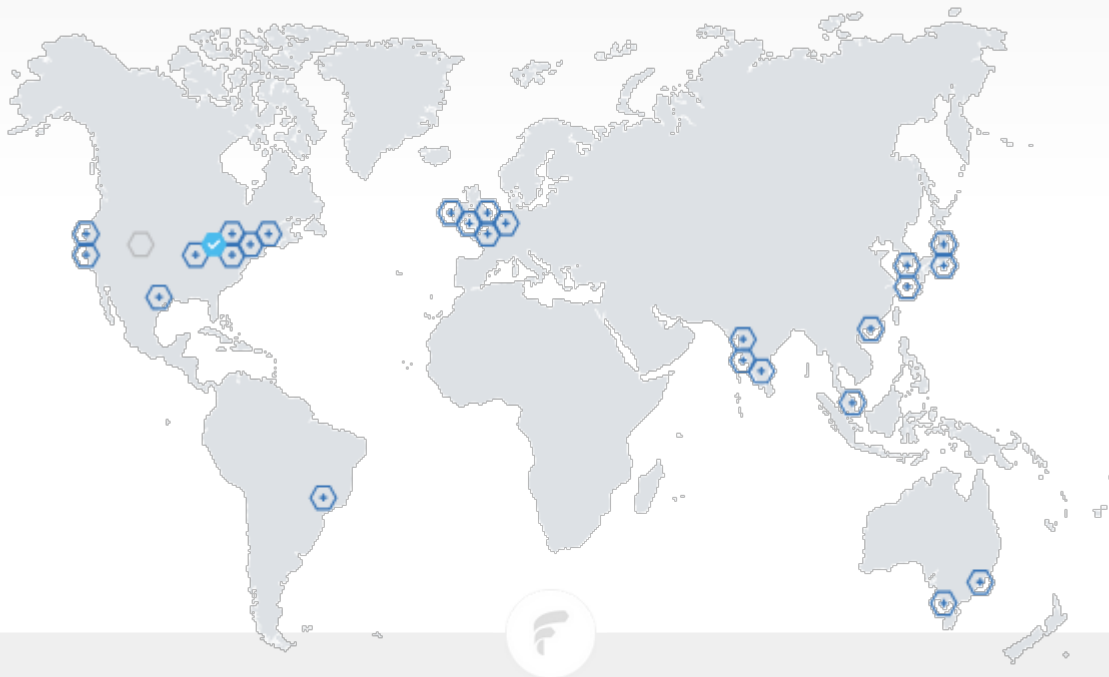
---

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



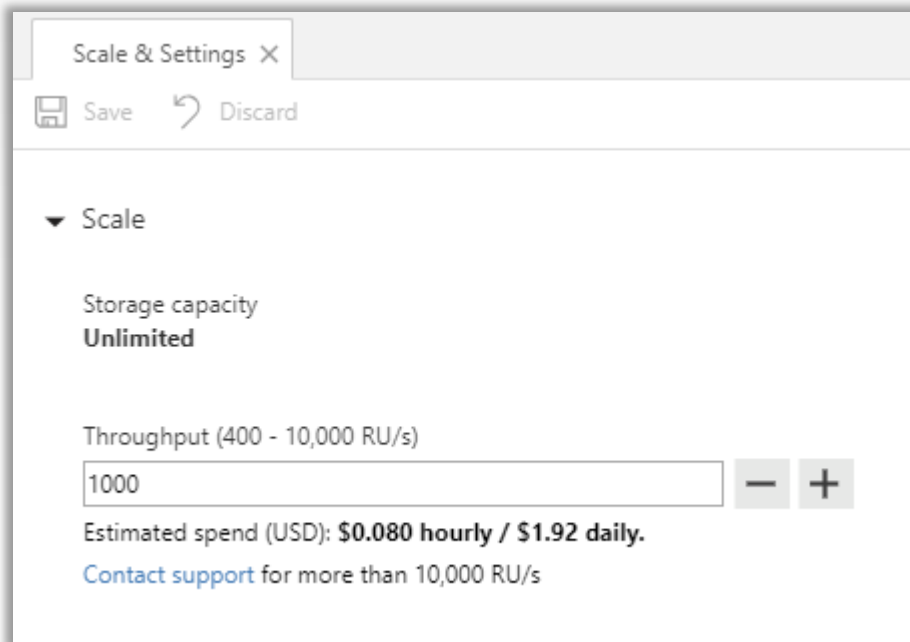
# What Is Azure Cosmos DB?

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



# What Is Azure Cosmos DB?

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



The screenshot shows the 'Scale & Settings' panel for an Azure Cosmos DB account. At the top, there is a tab labeled 'Scale & Settings' with a close button (X). Below the tab are two buttons: 'Save' (with a floppy disk icon) and 'Discard' (with a curved arrow icon). The main section is titled 'Scale' with a downward arrow. It contains two settings: 'Storage capacity' set to 'Unlimited' and 'Throughput (400 - 10,000 RU/s)' set to '1000'. The throughput setting has a text input field and two buttons, '-' and '+', for adjustment. Below the throughput setting, it shows the 'Estimated spend (USD): \$0.080 hourly / \$1.92 daily.' and a link to 'Contact support for more than 10,000 RU/s'.

Scale & Settings X

Save Discard

▼ Scale

Storage capacity  
**Unlimited**

Throughput (400 - 10,000 RU/s)

1000 - +

Estimated spend (USD): **\$0.080 hourly / \$1.92 daily.**

[Contact support](#) for more than 10,000 RU/s



# What Is Azure Cosmos DB?

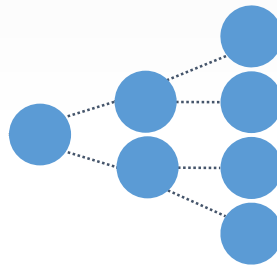
A globally distributed, massively scalable, multi-model database service



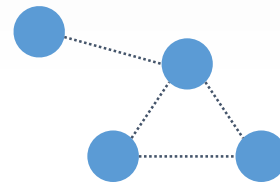
Key-Value



Column-Family



Document



Graph



# What Is Azure Cosmos DB?

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

Has multiple APIs as well



Table API



SQL



MongoDB



Coming Soon:



ANSI SQL



# What Is Azure Cosmos DB?

---

A database service featuring an engine built to excel at several things, but especially:

- Partitioning

- Replication



# What Is Azure Cosmos DB?

---

It's a NoSQL offering!

**3 DBAS WALKED INTO  
A NOSQL BAR....**

**A WHILE LATER THEY  
WALKED OUT BECAUSE THEY  
COULDN'T FIND A TABLE**

# What Is Azure Cosmos DB?

---

- I often hear NoSQL == No Schema == No Design
  - Not True
- GENERALLY NoSQL schemas
  - Do Exist
  - Are somewhat enforced by the database
  - Are fully enforced by the application
- There are still design decisions that need to happen early on
  - (And if they're wrong you will pay for it later)



# History of Azure Cosmos DB

- Microsoft started having problems with internal large scale apps
  - 2010 – “Project Florence”
  - 2014 – Azure DocumentDB
  - 2017 – Azure Cosmos DB
- MS leverages this internally
- Designed for the cloud
- One of the fastest-growing services on Azure



# Another Scenario

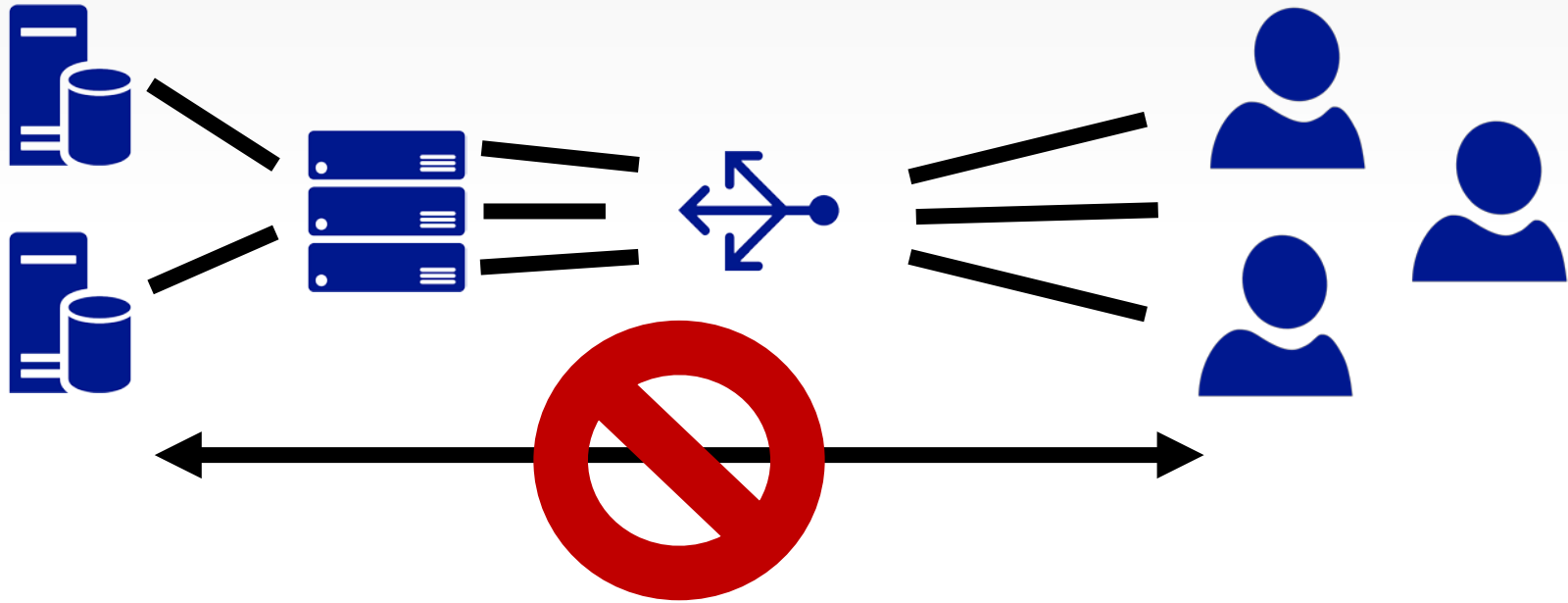
---

- We're developing an internal app for a global company
- Thousands of users reading/updating data
- How would we architect this?

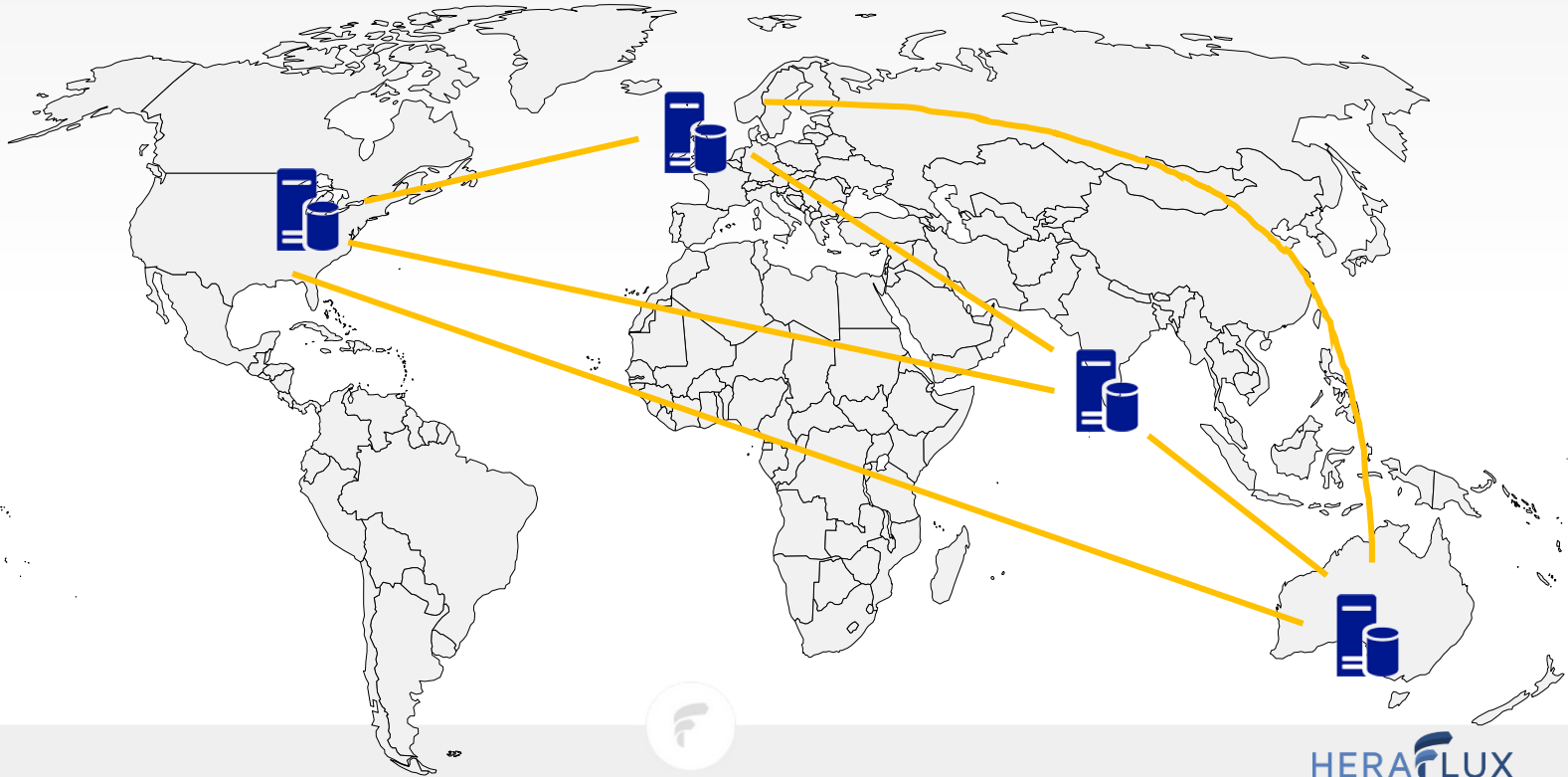


# Solution: SQL Server

---

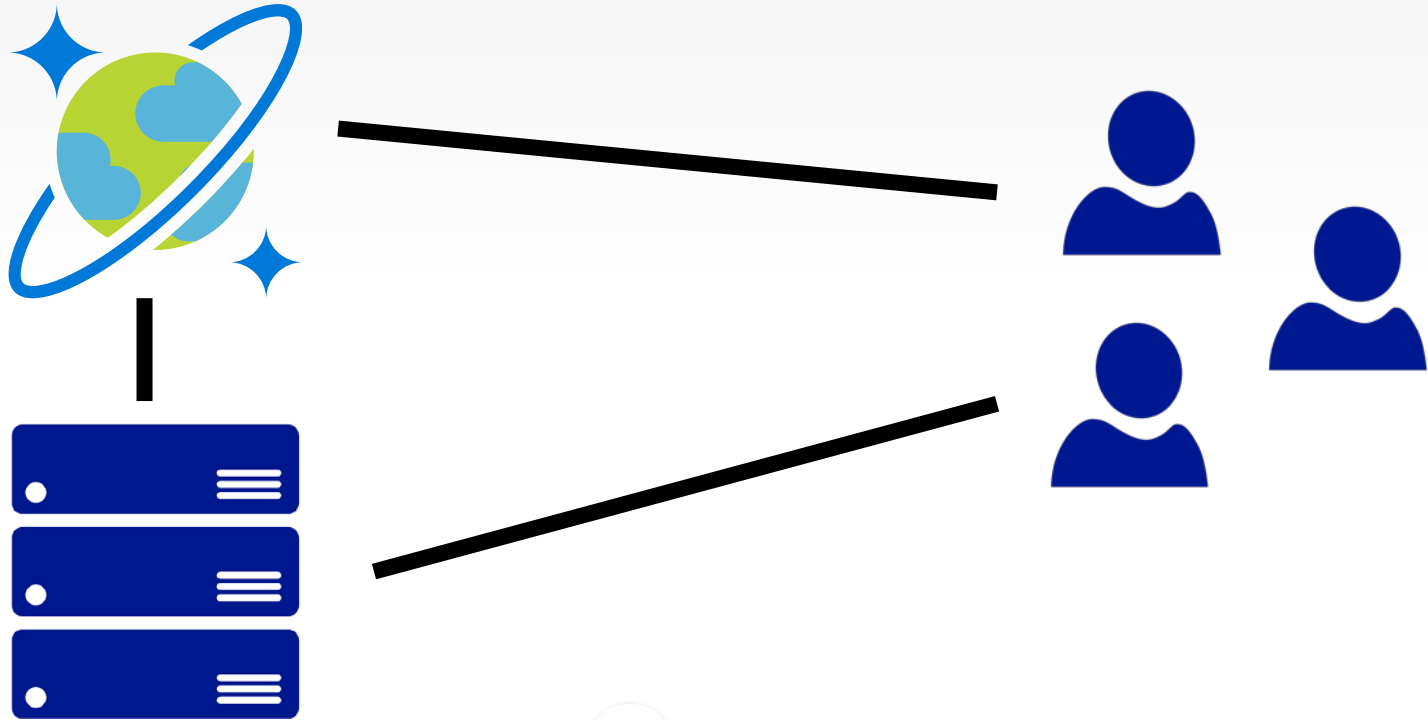


# Solution: SQL Server



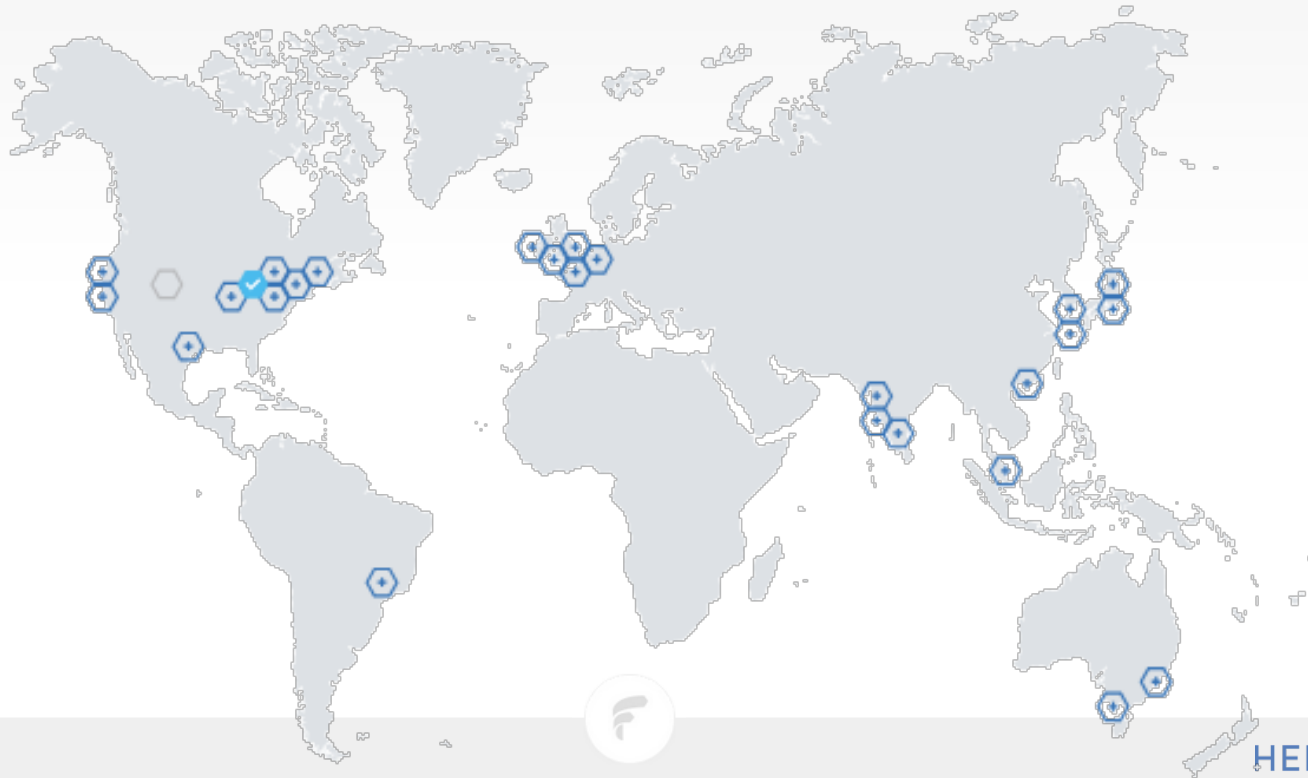
# Solution: Cosmos DB

---



# Solution: Cosmos DB

---

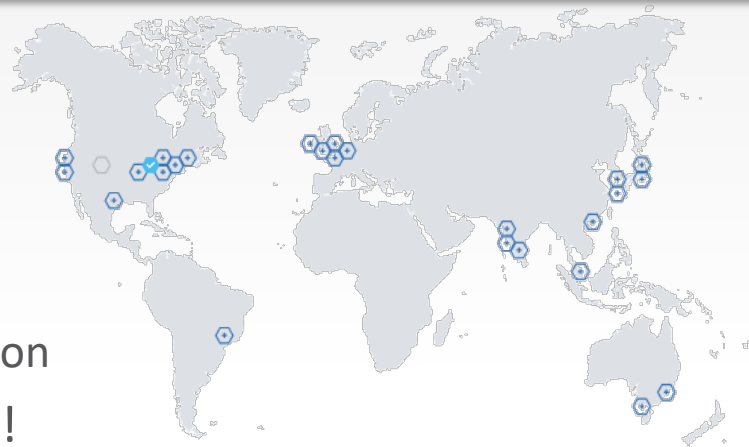




# What's It Offer?

# Fast Data Distribution

- Cosmos DB is a Foundational Azure service
- Put your data where the users are
- Replication between regions is automatic
- Multi-homing APIs
  - Clients automatically connect to the nearest region
- Adding or removing regions? No code changes!
- Manual or automatic failovers
- Designed from the ground up for HA



# Elastically Scalable

---

- Both storage and throughput can be scaled transparently
- A single machine is never a bottleneck
- Collections can scale from GB to PB across many machines and regions



# Guaranteed Low Latency

- Requests are served from the nearest region
- Database engine optimized for writes, latch-free
- Indexing is synchronous and automatic
- Single-digit millisecond read latency at 99<sup>th</sup> Percentile

	Reads (1KB)	Indexed Writes (1KB)
50 <sup>th</sup> Percentile	< 2ms	< 6ms
99 <sup>th</sup> Percentile	< 10ms	< 15ms



# Guaranteed Availability

---

- 99.99% availability when in a single region
- 99.999% availability in multiple regions
- Highly-redundant storage architecture
- Automatic or manual failover



# What do these guarantees mean?

---



# Encryption

---

- All data is encrypted, period.
- In transit and at rest



# Permissions

- Two types of keys:
- Master Keys
  - Administrative
  - Grant access to the entire account (not granular)
  - Read-write and Read-only

Read-write Keys

[Read-only Keys](#)

URI

`https://sqlbobbdemo1.documents.azure.com:443/`



PRIMARY KEY

`https://sqlbobbdemo1.documents.azure.com:443/keys/primarykey/`



SECONDARY KEY

`https://sqlbobbdemo1.documents.azure.com:443/keys/secondarykey/`



PRIMARY CONNECTION STRING

`AccountKey=primarykey;AccountName=sqlbobbdemo1;DatabaseName=sqlbobbdemo1;Host=sqlbobbdemo1.documents.azure.com;Password=;Server=tcp:sqlbobbdemo1.documents.azure.com,1433;User Id=sqlbobbdemo1;Version=12.0`



SECONDARY CONNECTION STRING

`AccountKey=secondarykey;AccountName=sqlbobbdemo1;DatabaseName=sqlbobbdemo1;Host=sqlbobbdemo1.documents.azure.com;Password=;Server=tcp:sqlbobbdemo1.documents.azure.com,1433;User Id=sqlbobbdemo1;Version=12.0`



# Permissions

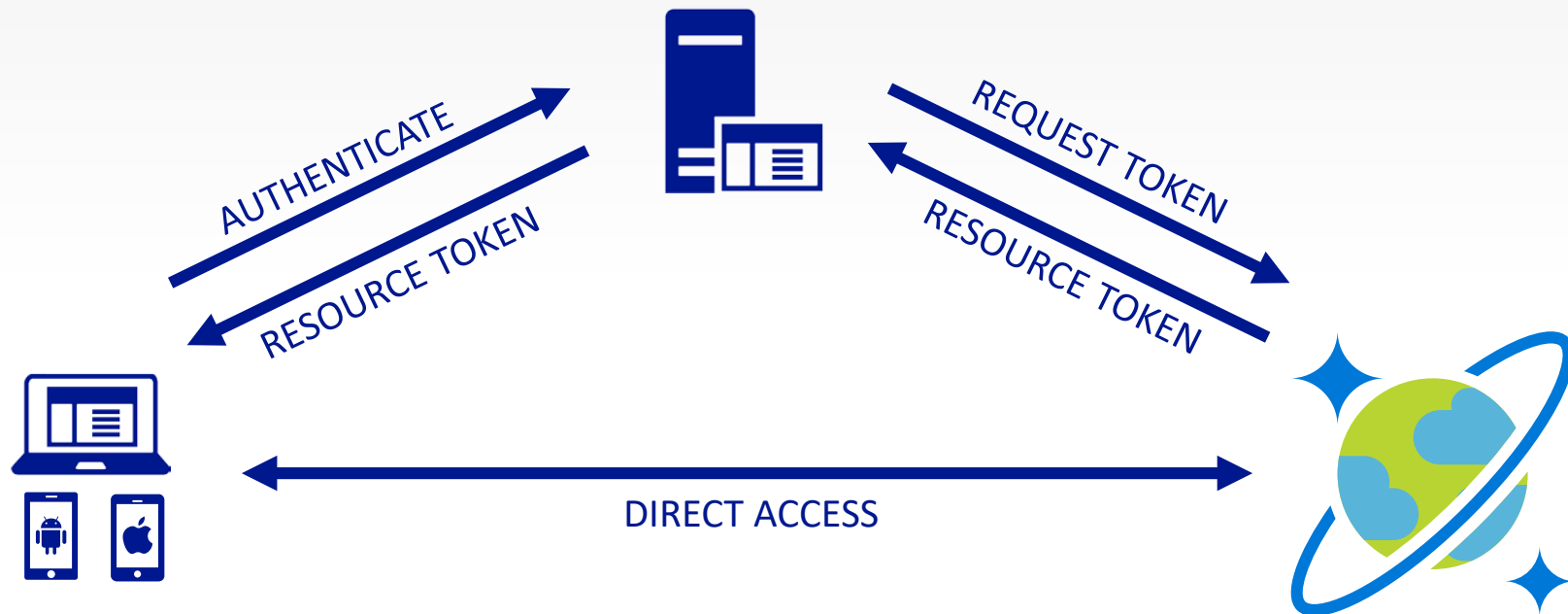
---

- Resource Tokens
- Used for application resources (Containers, docs, SPs, Triggers, UDF, etc.)
  - Kinda like SQL Permissions
- Tokens are specific to {user, resource, permission}
- Tokens are time-sensitive (default 1 hour, max 5 hours)



# Permissions

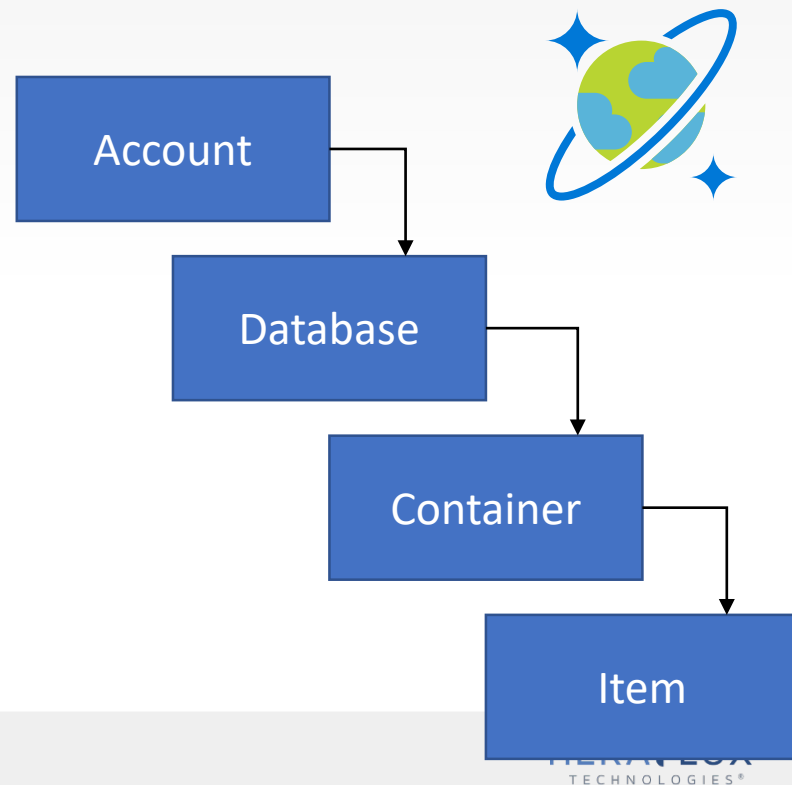
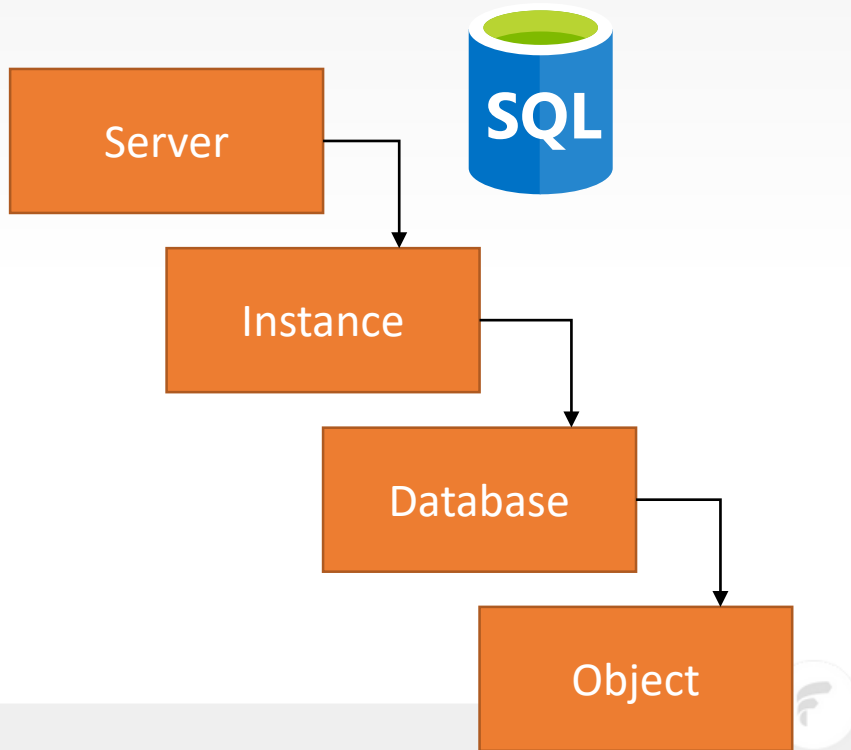
- Resource Tokens



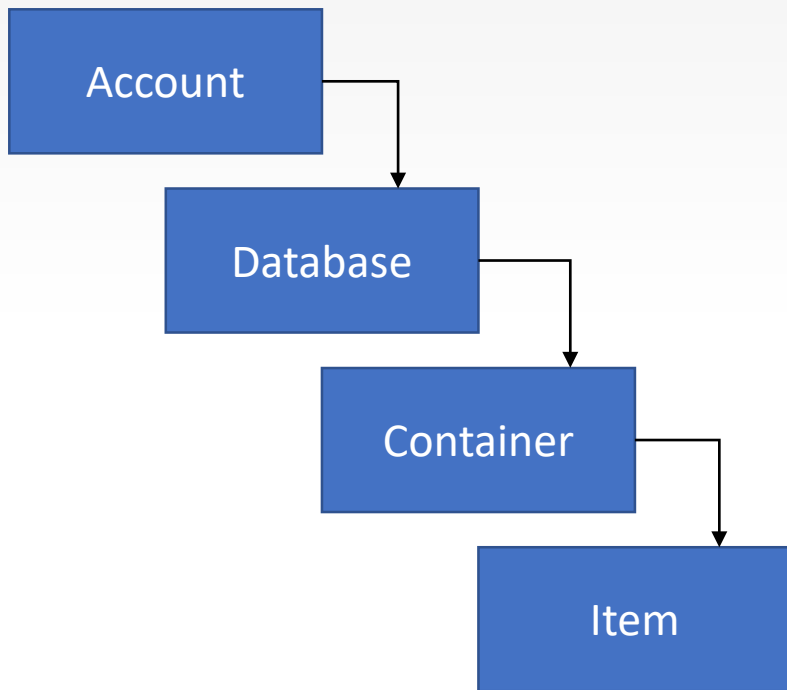
# How Does It Work?



# Resource Model



# Resource Model



Home > Azure Cosmos DB > Azure Cosmos DB

## Azure Cosmos DB

New account

\* ID

documents.azure.com

\* API ⓘ

▼

\* Subscription

▼

\* Resource Group

☒ Create new ☐ Use existing

\* Location

▼

☐ Enable geo-redundancy ⓘ

☐ Enable Multi Master ⓘ

---

Multi Master Preview

[Sign up to preview today](#) >

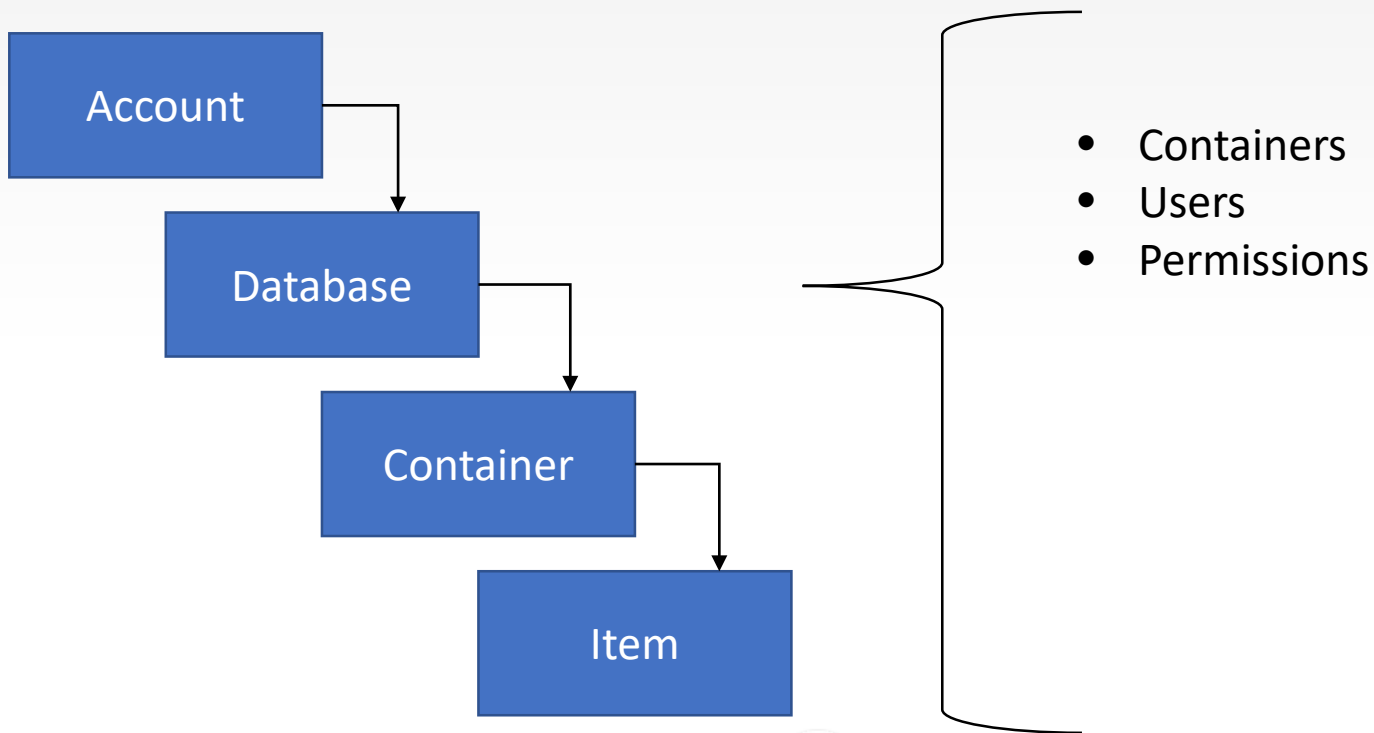
---

Virtual networks

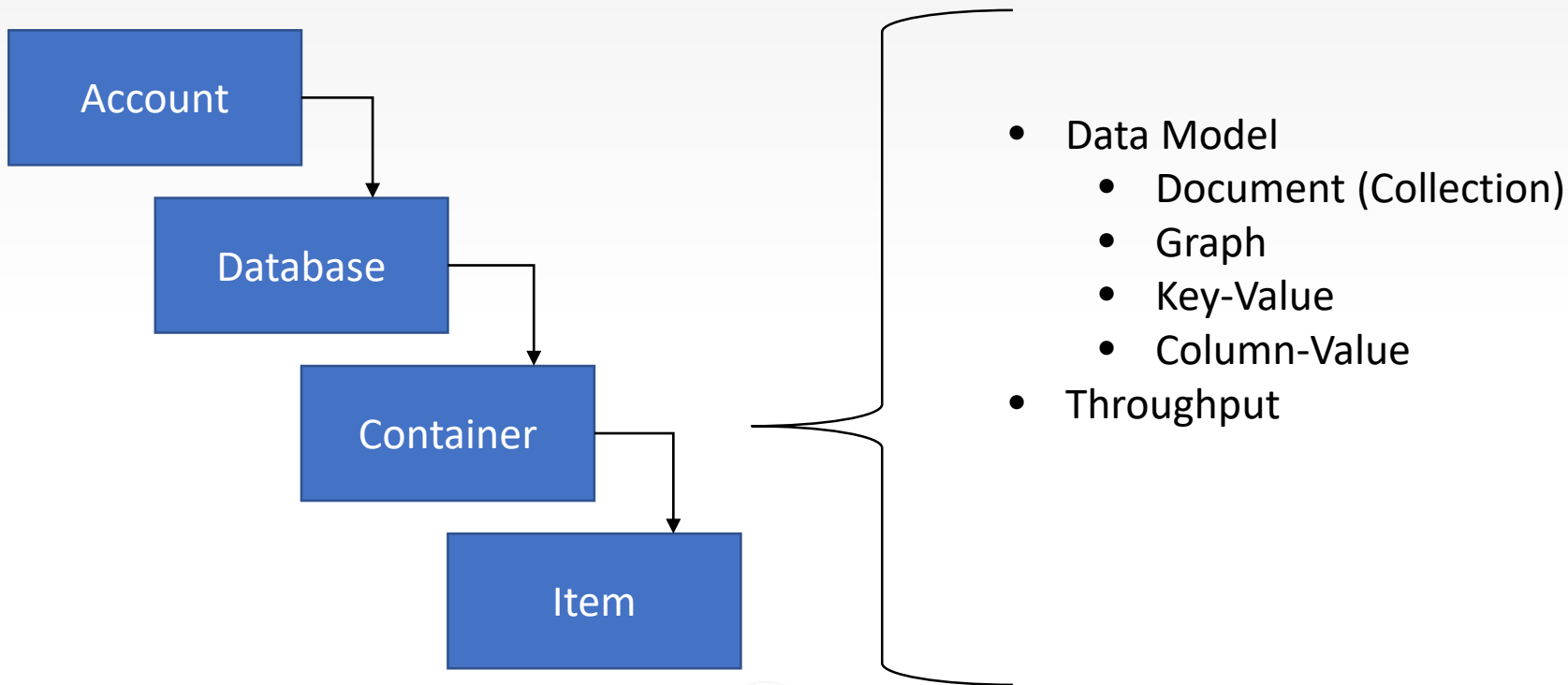
Configure virtual networks ⓘ



# Resource Model



# Resource Model



# Resource Model

## ATOM RECORD SEQUENCE (ARS) SYSTEM

Atoms = primitives (string, bool, etc)

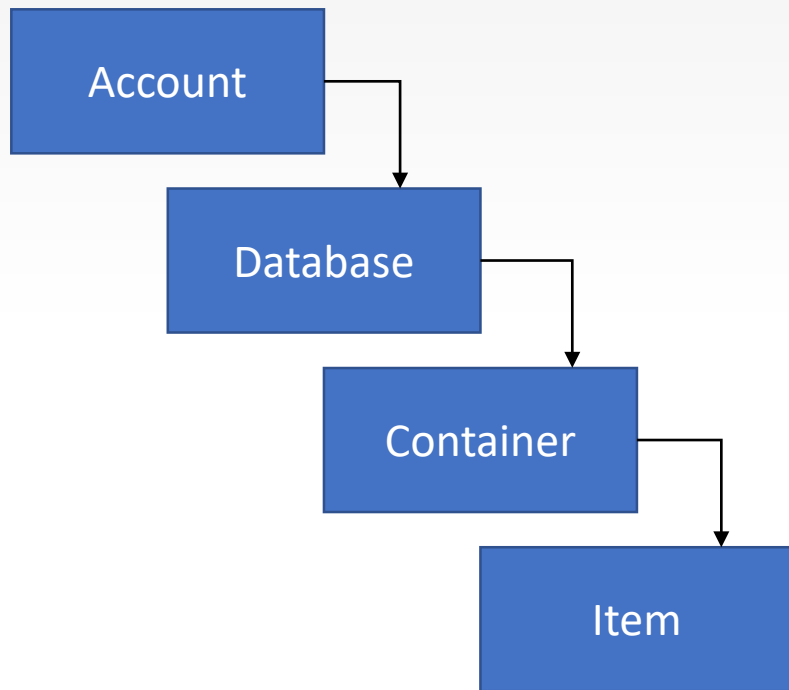
Records = structs of atoms

Sequences = arrays of {atom, record, sequence}

Cosmos DB translates & projects all data models into an ARS model

- Data Model
  - Document (Collection)
  - Graph
  - Key-Value
  - Column-Value
- Throughput

# Resource Model



×

Add Collection

\* Database id ⓘ

☒ Create new ☐ Use existing

☐ Provision database throughput ⓘ

\* Collection Id ⓘ

\* Storage capacity ⓘ

Fixed (10 GB)

Unlimited

\* Partition key ⓘ

\* Throughput (1,000 - 100,000 RU/s) ⓘ

−

+

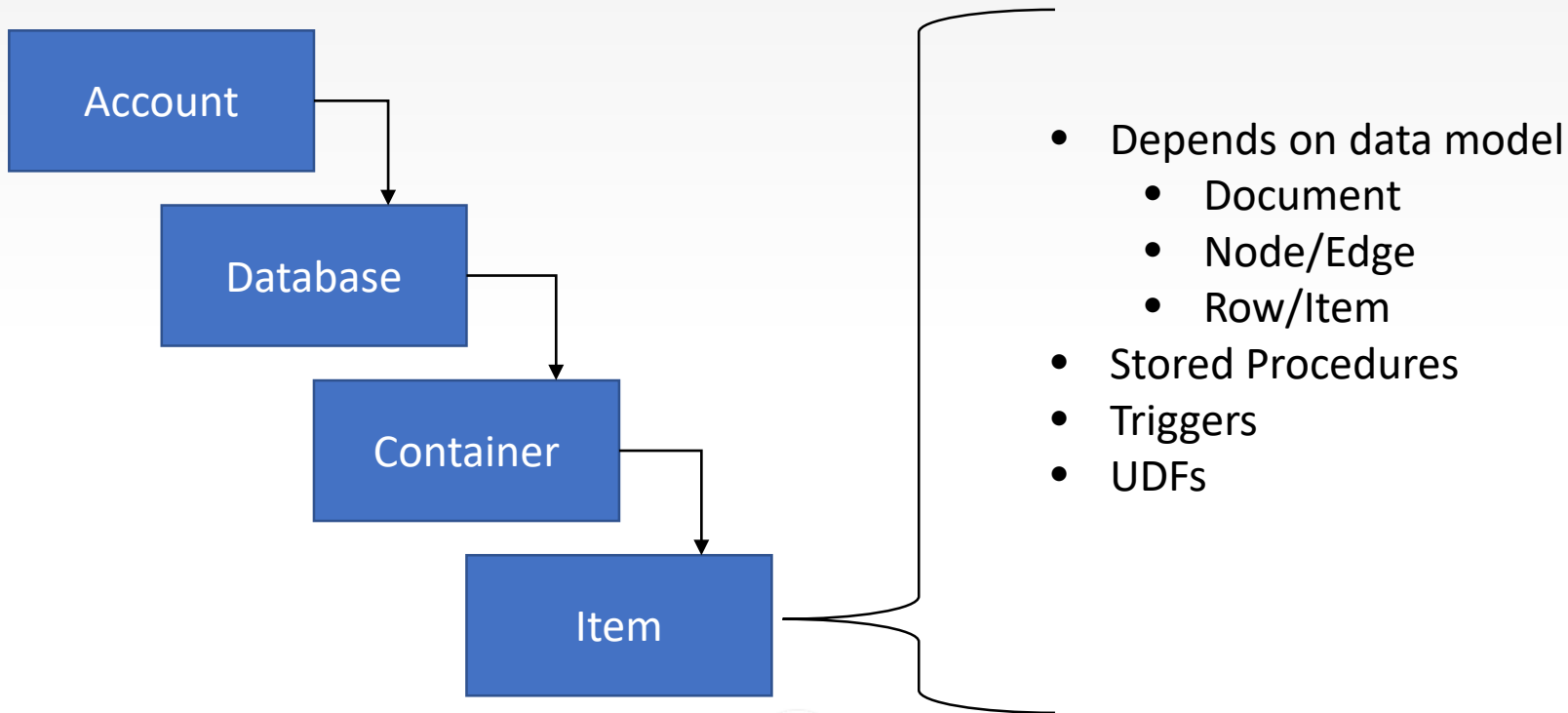
[Contact support](#) for more than 100,000 RU/s.

Unique keys ⓘ

+ Add unique key



# Resource Model



# System Architecture

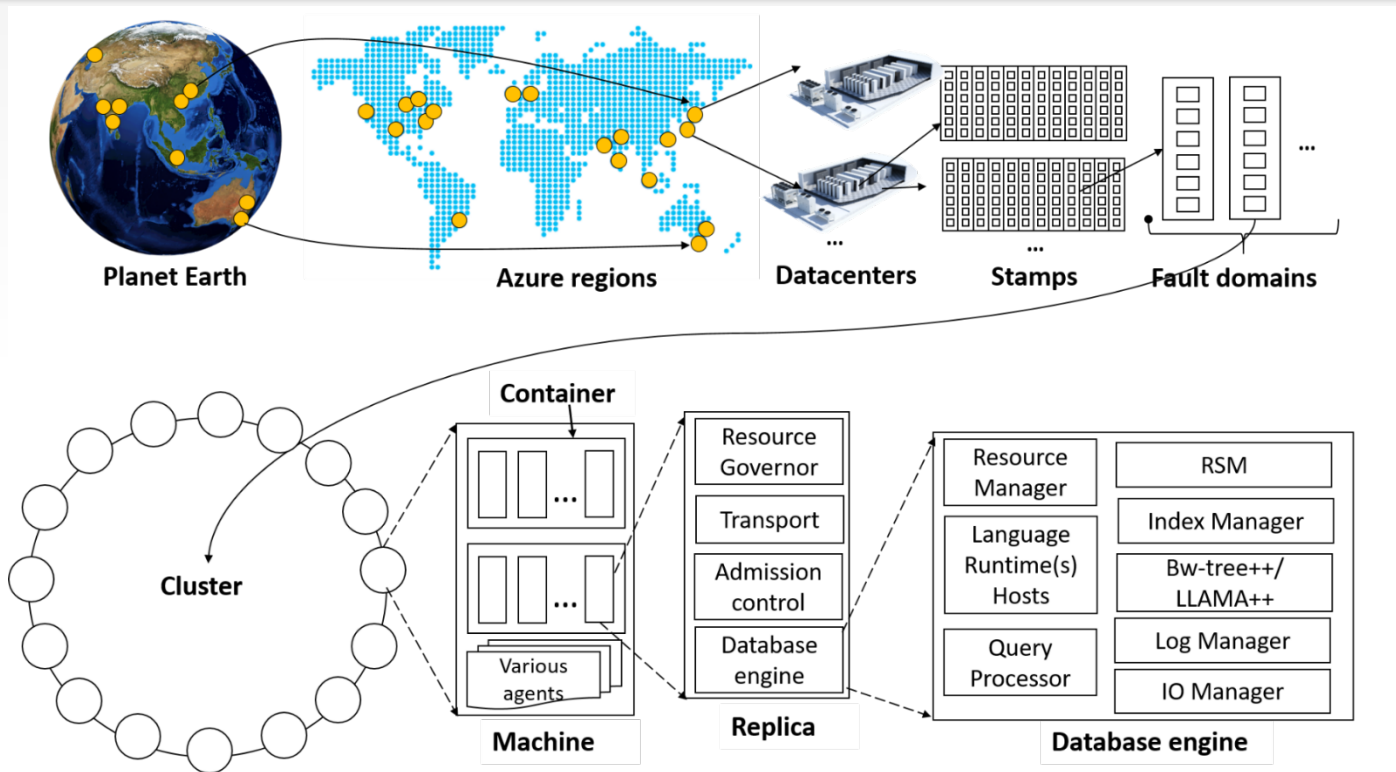


Image: Microsoft



# Request Units

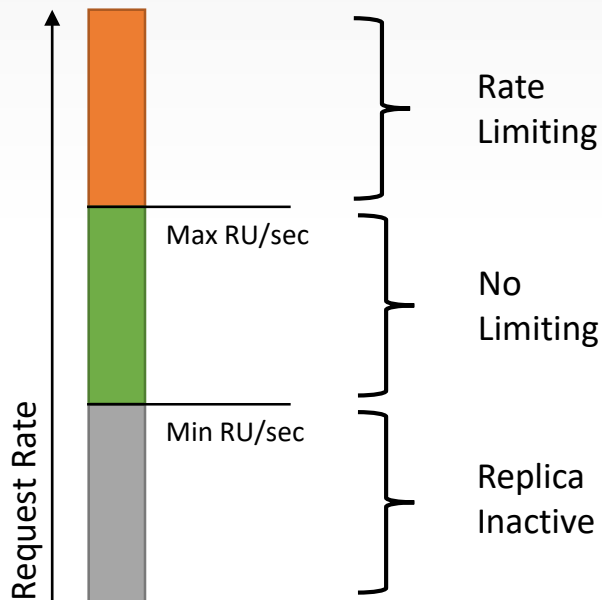
---

- RU is the rate-based currency of Cosmos DB
- Represents a combination of CPU, Memory, and IO
- 1 RU = 1 read of 1KB
- Every request is assigned a “cost” in RUs
  - Reads, writes, stored procedures, etc



# Request Units

- Provisioned in units of RU/second
- Can be changed at any time; metered hourly
- Exceeding your RU budget = rate limiting
- When inactive, background tasks run
  - Index Maintenance
  - TTL Expiration



# Partitioning



# Partitioning in SQL Server

---

- Define boundary values between partitions
- Map partitions to physical locations (filegroups)
- Similar values generally in the same partition
  - Can lead to “hot” partitions
  - Especially if on dates
- Partition management is manual
- Hard Limit: 15.000 partitions per table



# Partitioning in Cosmos DB

---

- There are no “ranges”, every partition key is hashed
- Logical partitions (keys) are spread across physical partitions
- Partition management is automatic!
- No limit on number of partitions
- Hard limit: 10GB max of data per partition key



# Partitioning

---

- The most important design decision in Cosmos DB
- Has a direct effect on
  - How well it will scale
  - How much you will pay
- Think through partitioning during the design phase, it's easier!



# Partitioning

Add Collection

\* Database id ⓘ

☒ Create new

☐ Use existing

Type a new database id

☐ Provision database throughput ⓘ

\* Collection Id ⓘ

e.g., Collection1

\* Storage capacity ⓘ

Fixed (10 GB)

Unlimited

\* Partition key ⓘ

e.g., /address/zipCode

\* Throughput (1,000 - 100,000 RU/s) ⓘ

-

+

Contact support for more than 100,000 RU/s.

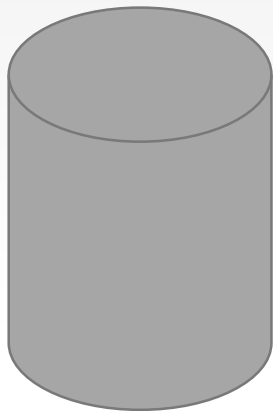
Unique keys ⓘ

+ Add unique key

HERAFLUX  
TECHNOLOGIES®

# Partitioning

---

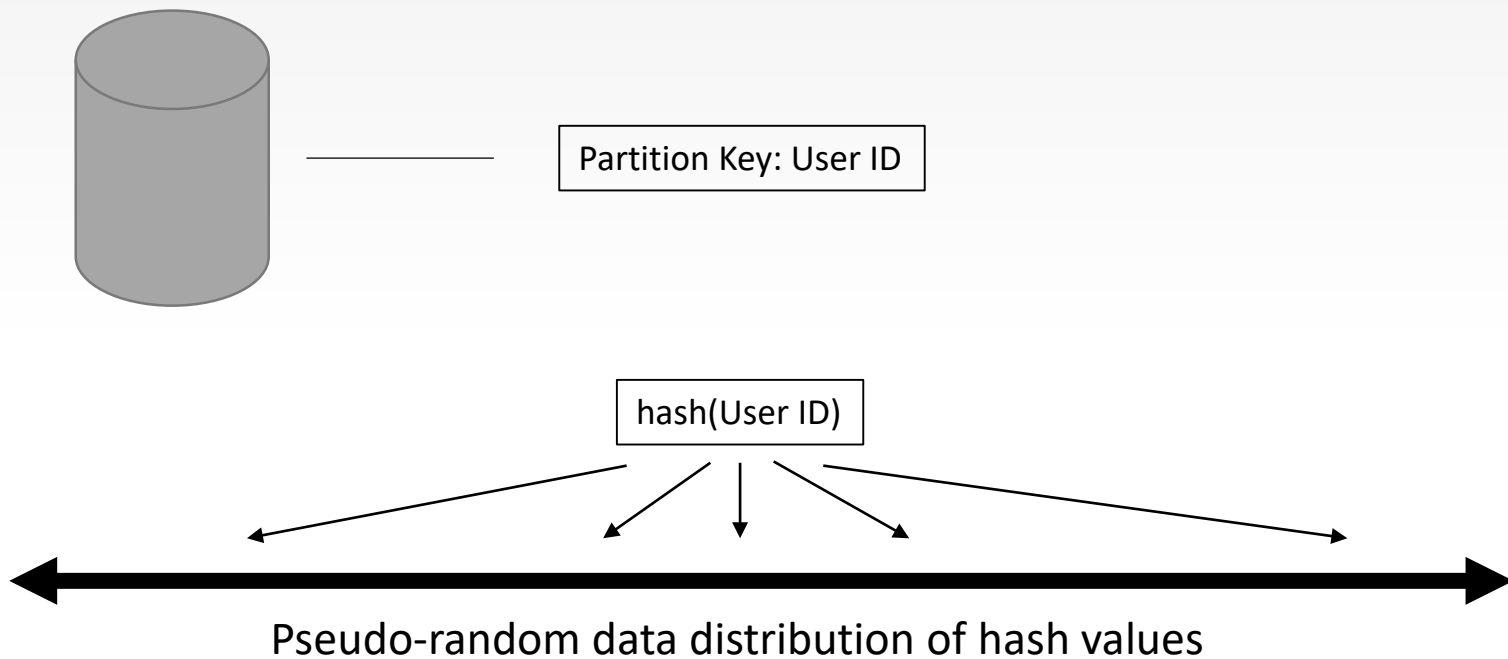


Cosmos DB Container

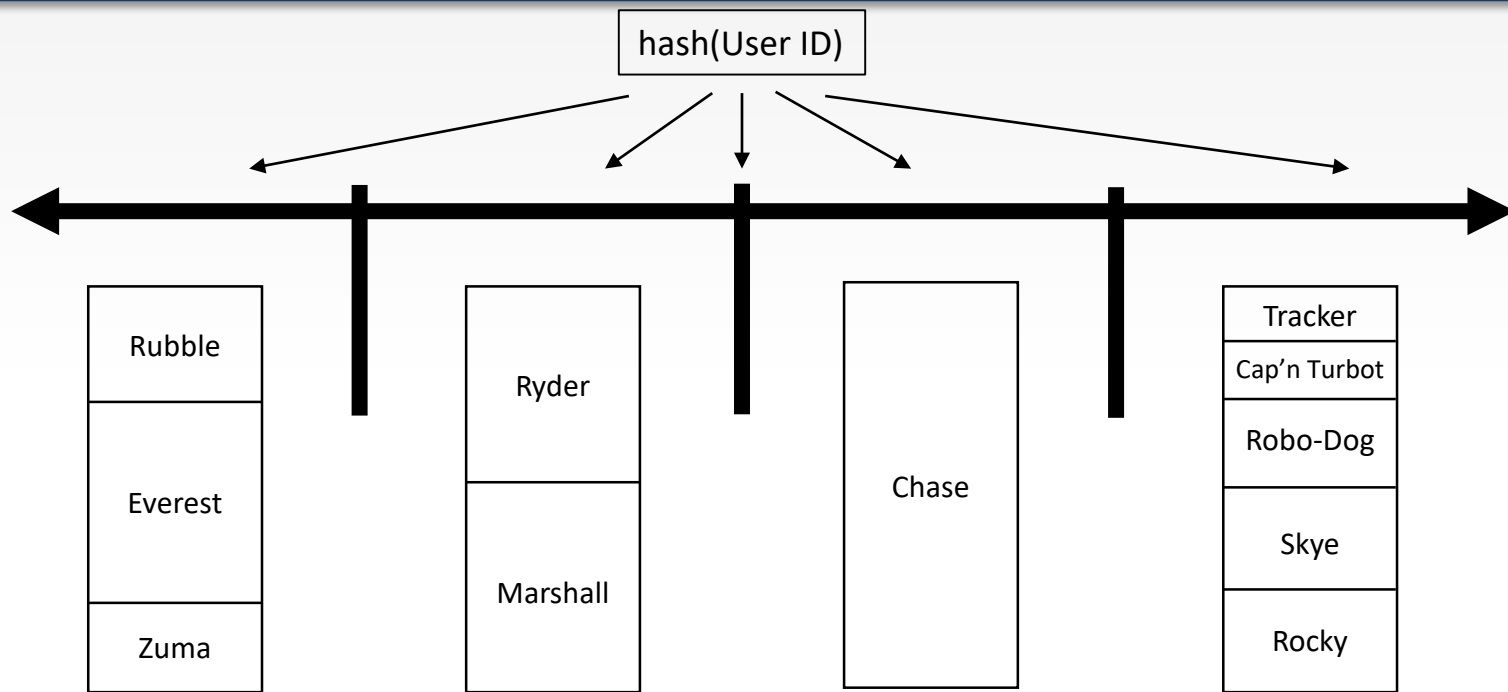
Partition Key: User ID



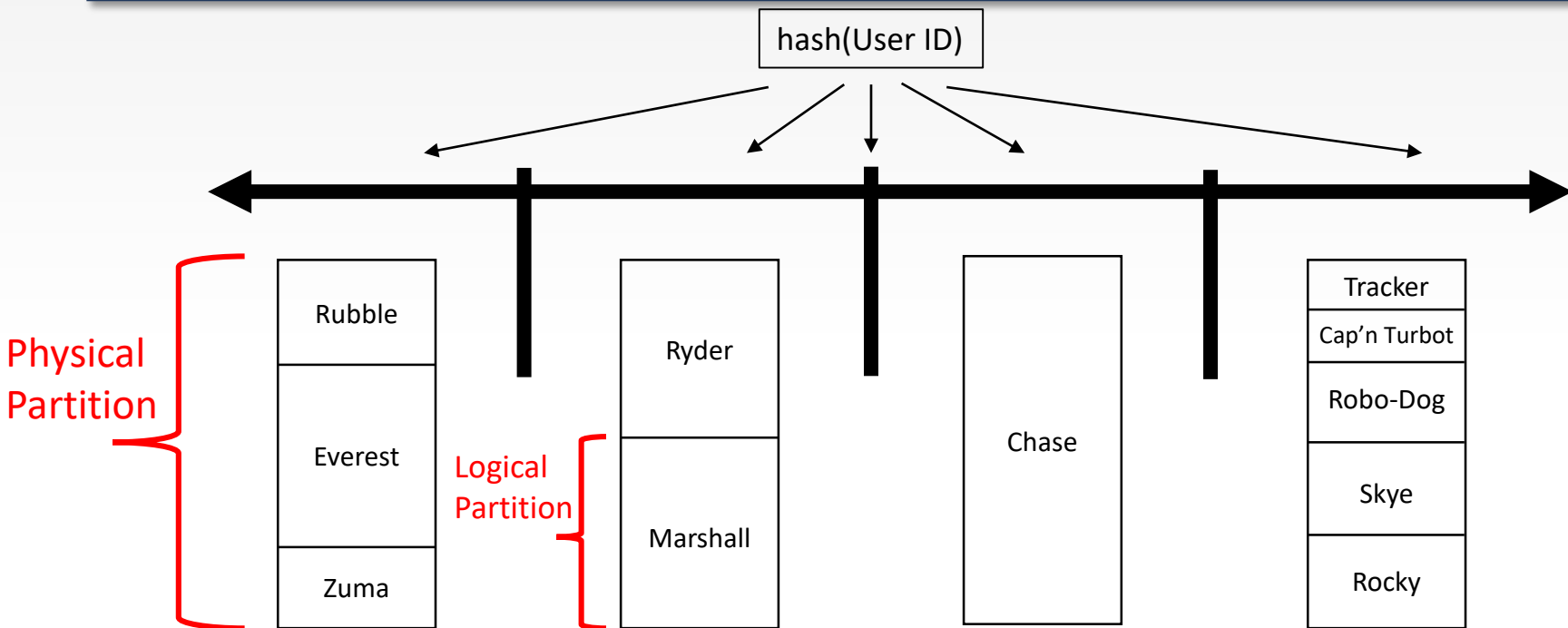
# Partitioning



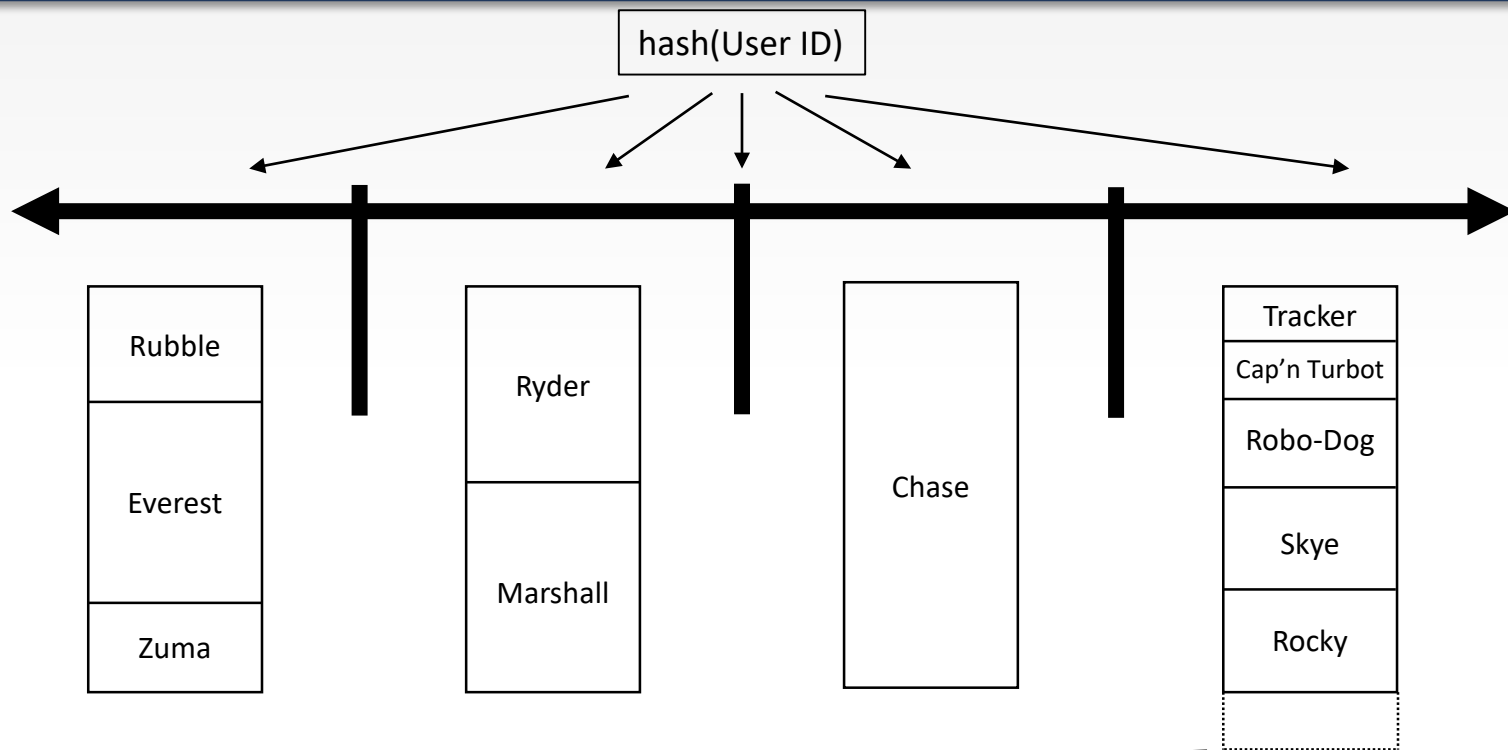
# Partitioning (Physical Partition Sets)



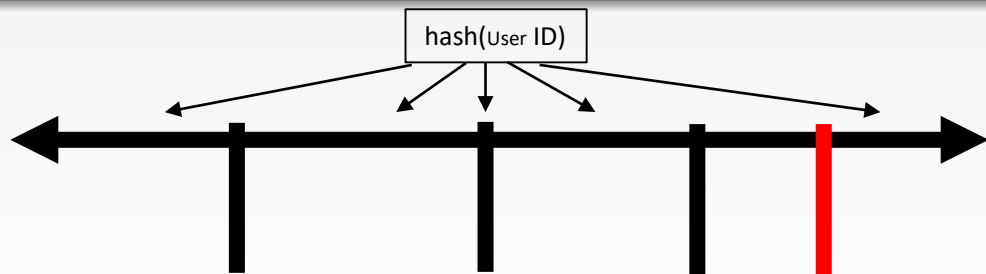
# Partitioning (Physical Partition Sets)



# Partitioning (Physical Partition Sets)

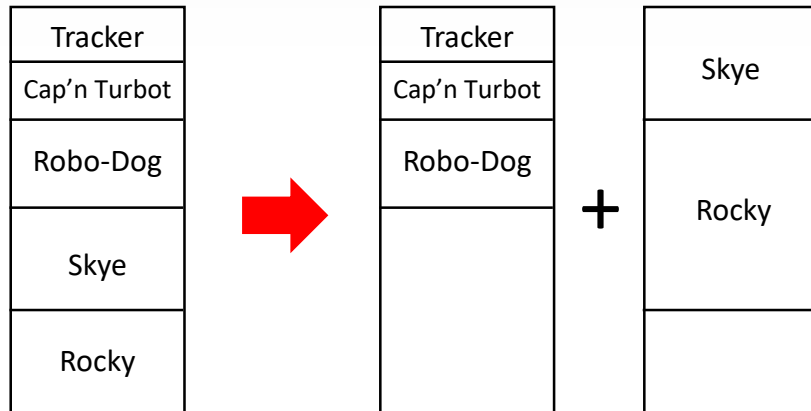


# Partitioning (Physical Partition Sets)



Partitions can be dynamically subdivided to grow the database without affecting availability

This is done automatically.



# Choosing A Good Partition Key

---

- Plan to distribute both request and storage volume
  - Remember the 10GB limit
  - Adding dates after partition values can help with this
- For greatest efficiency, queries should eliminate partitions
- Queries can be routed/filtered via partition key
- “Fan-Out” is something to try to avoid where possible



# Choosing A Good Partition Key

---

- Understand your workload!
- Understand the most frequent/expensive queries
- Understand insert vs update ratios
- Remember partition keys are logical!
  - Don't be afraid of having too many
  - More key values = better scalability



# Consistency



# Consistency

---

- This is huge because we have multiple replicas
- If a change is replicated, what is seen elsewhere?
- Why replicate, anyway?
  - HA – multiple copies for failover
  - Speed!
- Bring the data closer to the user
- “cheat” the speed of light!



# Consistency: Containers and Replicas

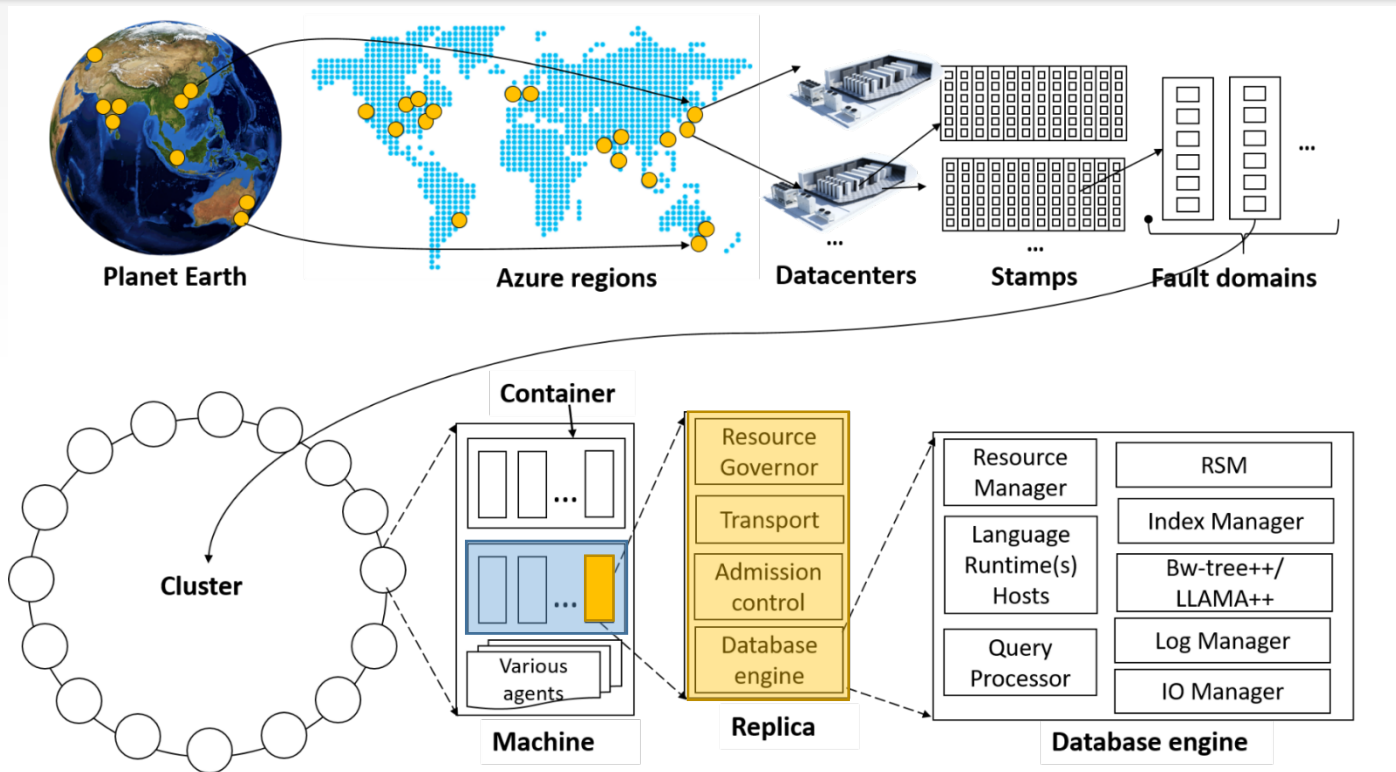
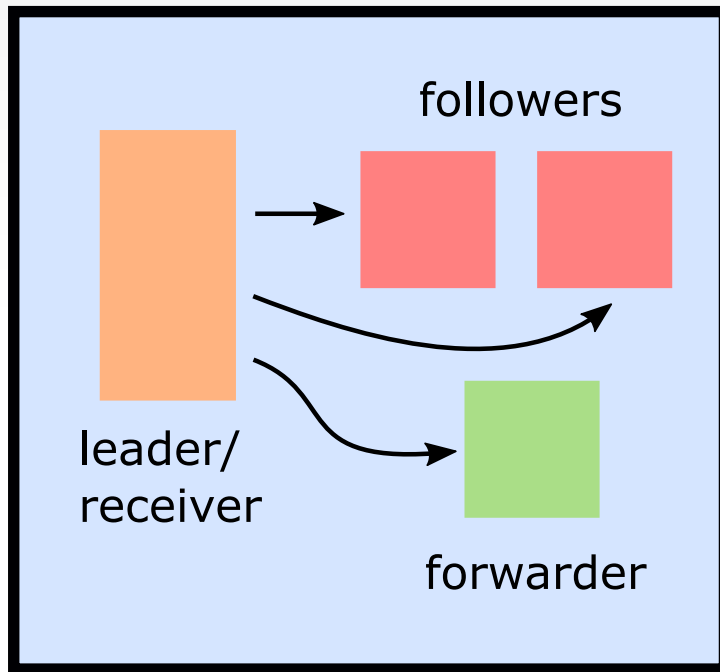


Image: Microsoft

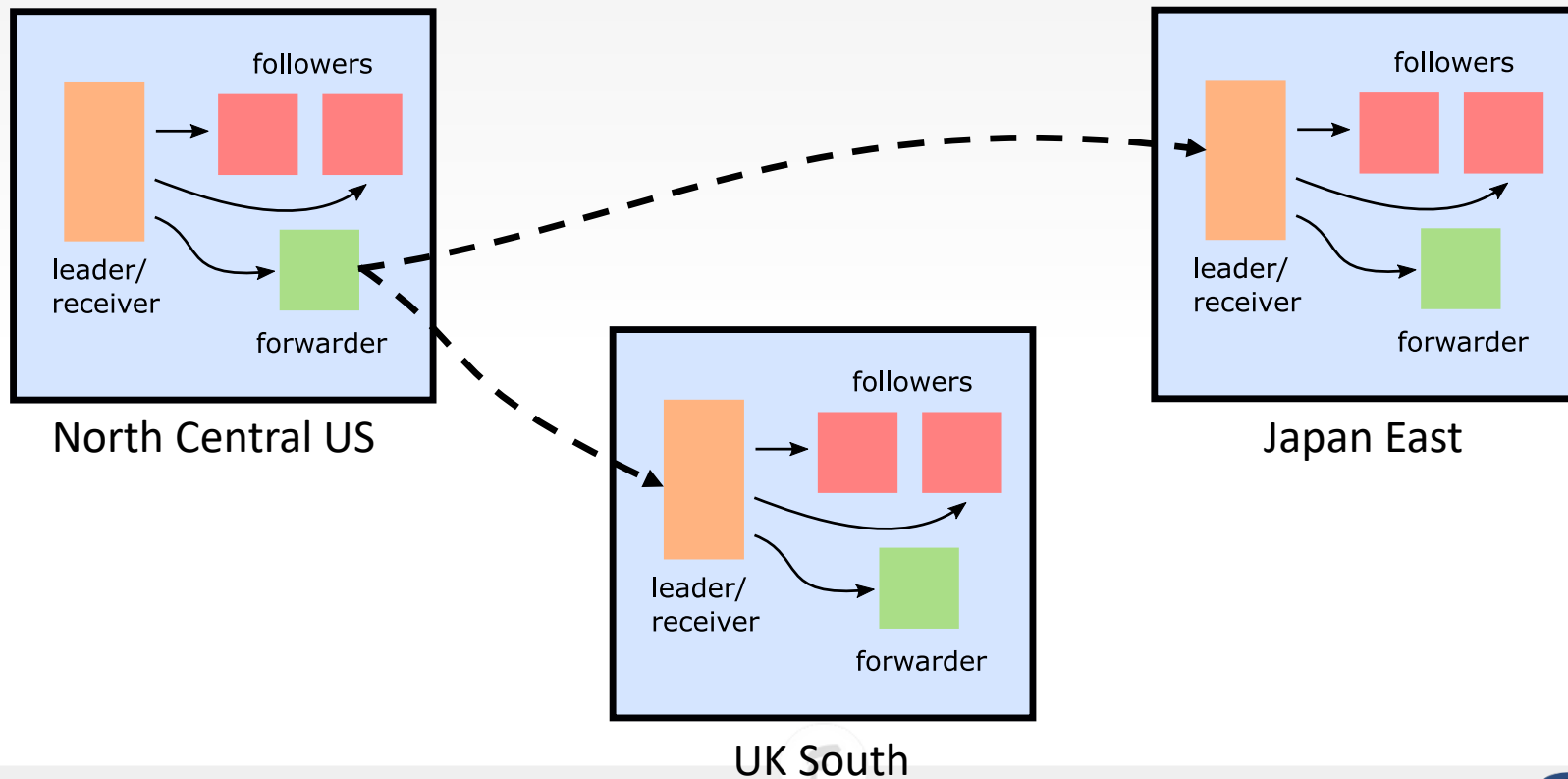


# Consistency: Replica Sets

---



# Consistency: Replica Sets



# Consistency: Models

- I love consistency models
- I also love isolation levels

Locks, Blocks, and Snapshots:  
Maximizing Database  
Concurrency

Bob Pusateri

HERAFLUX  
TECHNOLOGIES®

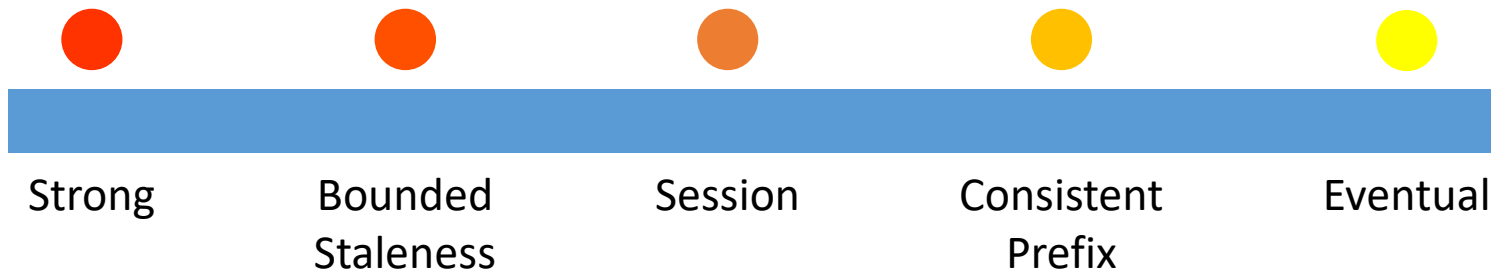


HERAFLUX  
TECHNOLOGIES®

# Consistency Models

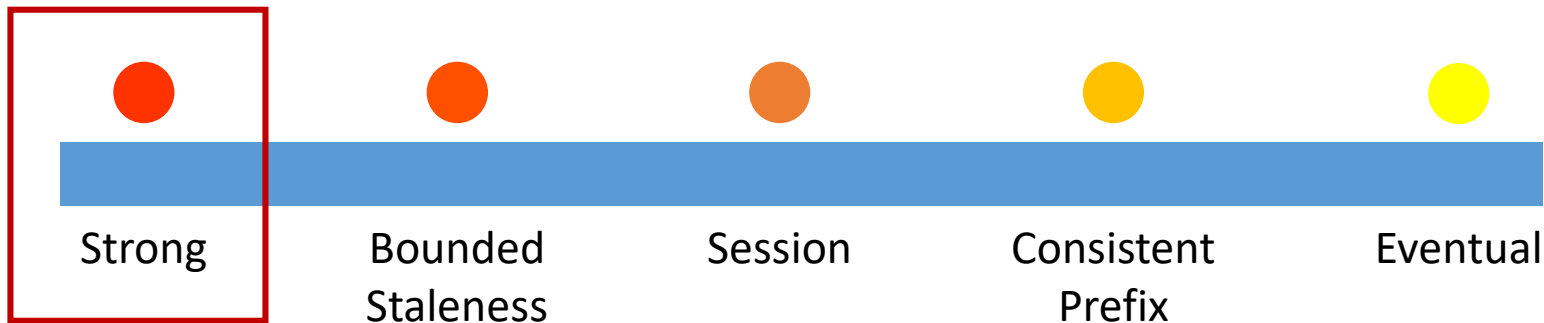
---

- Azure Cosmos DB has 5 of them
- You can choose what gets prioritized
- Can be overridden on a per-request basis



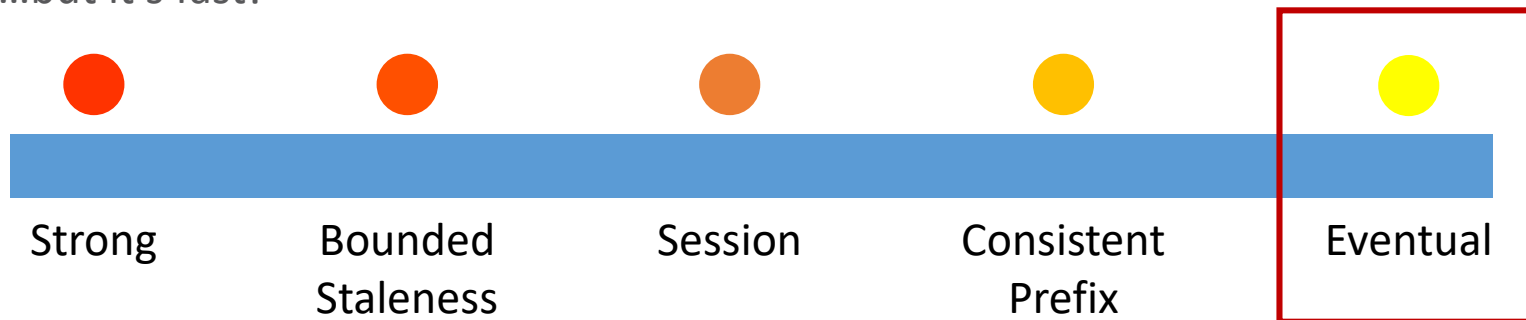
# Consistency Models: Strong

- Linearizability guarantee: reads will always return the most recent version of an item
- (Like SERIALIZABLE [maybe?])
- Writes are only visible after committed by a majority quorum of replicas



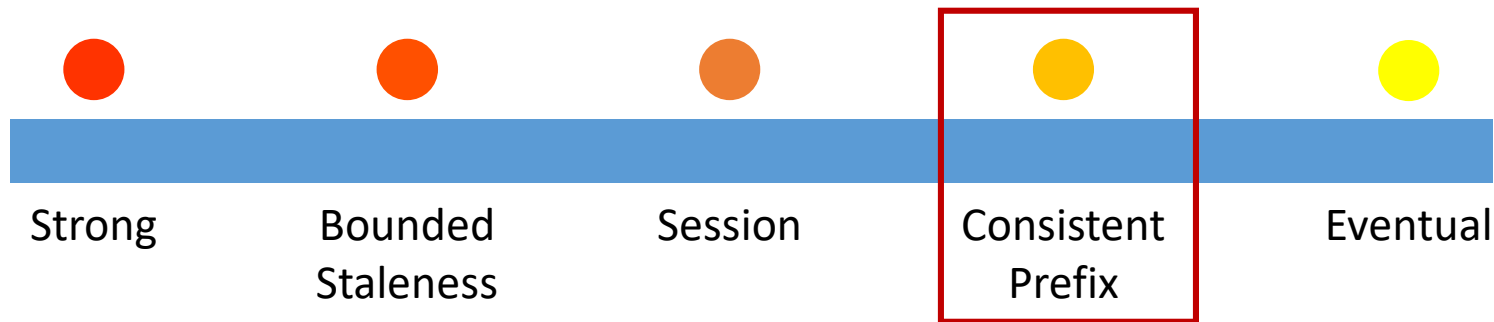
# Consistency Models: Eventual

- Guarantees that “absence of any further writes, replicas will eventually converge”
- No guarantee of order
  - Client may get “new” values older than ones it had previously seen
- Lowest latency for reads and writes
  - ...but it's fast!



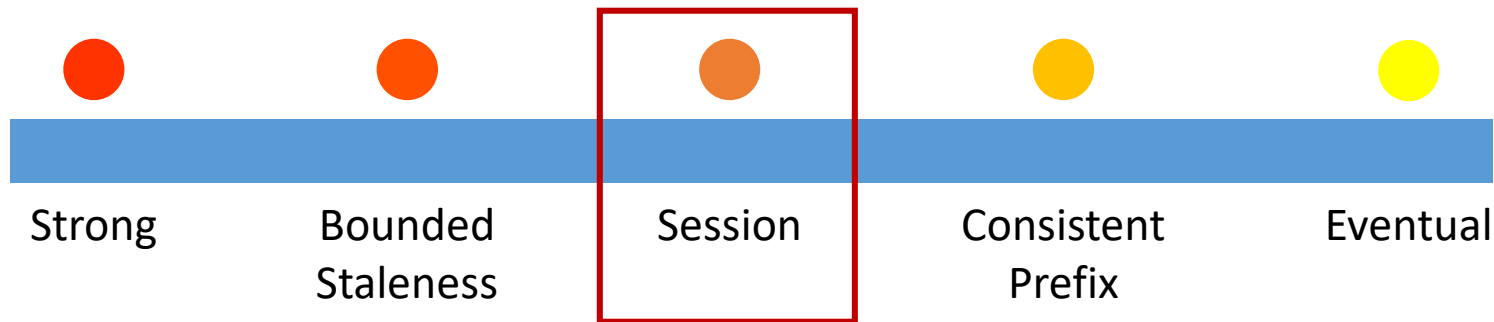
# Consistency Models: Consistent Prefix

- Guarantees that readers will always see writes in order



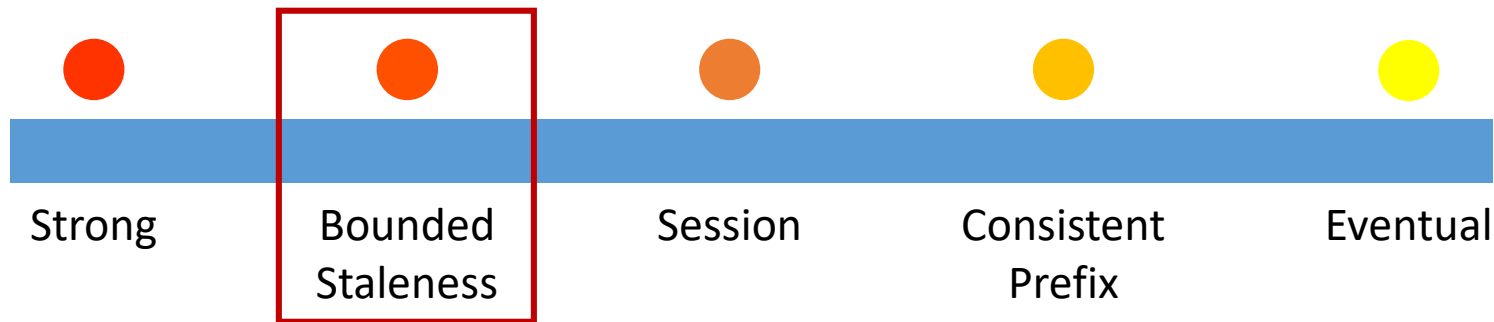
# Consistency Models: Session

- Scoped to a client session
  - There's a session key that is passed around
- Provides predictable consistency within a session
  - Monotonic reads & writes
  - Guarantee that you can read your own writes immediately
- Great predictability for your session, good performance for everyone else



# Consistency Models: Bounded Staleness

- Define a “window” of staleness in terms of # revisions or time
- If a replica gets too far behind (is outside the “window”)
  - Cosmos DB will prioritize consistency over all else
  - May even rate limit writes until stale replica catches up



# Consistency Models

[Home](#) > [sqlboddemo1 - Default consistency](#)

sqlboddemo1 - Default consistency  
Azure Cosmos DB account

Search (Ctrl+ /)


[Overview](#)  
[Activity log](#)  
[Access control \(IAM\)](#)  
[Tags](#)  
[Diagnose and solve problems](#)  
[Quick start](#)  
[Data Explorer](#)

SETTINGS

[Replicate data globally](#)  
[Default consistency](#)  
[Firewall and virtual networks](#)  
[Keys](#)  
[Add Azure Search](#)


Save Discard

STRONG BOUNDED STALENESS **SESSION** CONSISTENT PREFIX EVENTUAL

 Session consistency is most widely used consistency level both for single region as well as, globally distributed applications.

### Understand Session consistency

It provides write latencies, availability and read throughput comparable to that of eventual consistency but also provides the consistency guarantees that suit the needs of applications written to operate in the context of a user.

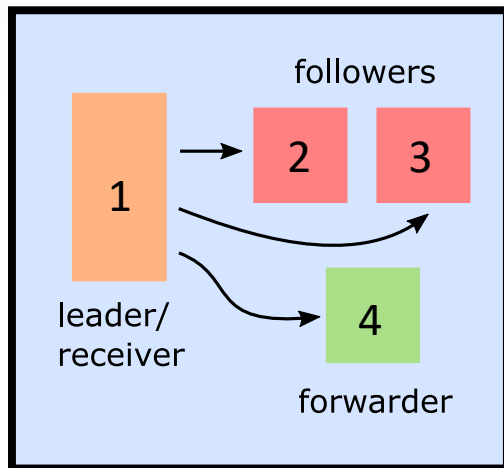


Region	Operation	Session	Notes
North Central US	Writes	Session A	Initial write
North Central US	Reads	Session A	Reads the same data as the write
Australia East	Reads	Session A	Reads the same data as the write
Australia Southeast	Reads	Session B	Reads the same data as the write



# Consistency Models

- What if you're not doing geo-replication? Does this matter?
- Yes it does!
- Even in local regions there are still 4 replicas of your database



# Indexing



# Indexing

---

- Yeah, about that....



# Indexing

---



# Indexing

---





# Indexing

**Recipes : Table**

	Field Name	Data Type	Description
▶	field1	Text	
	field2	Text	
	field3	Text	
	field4	Text	
	field5	Text	
	field6	Text	

Field Properties

General | **Lookup**

Field Size	50
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	Yes (Duplicates OK)
Unicode Compression	No

Yes (Duplicates OK)  
Yes (No Duplicates)

An index speeds up searches and sorting on the field, but may slow updates. Selecting "Yes - No Duplicates" prohibits duplicate values in the field. Press F1 for help on indexed fields.



# Indexing

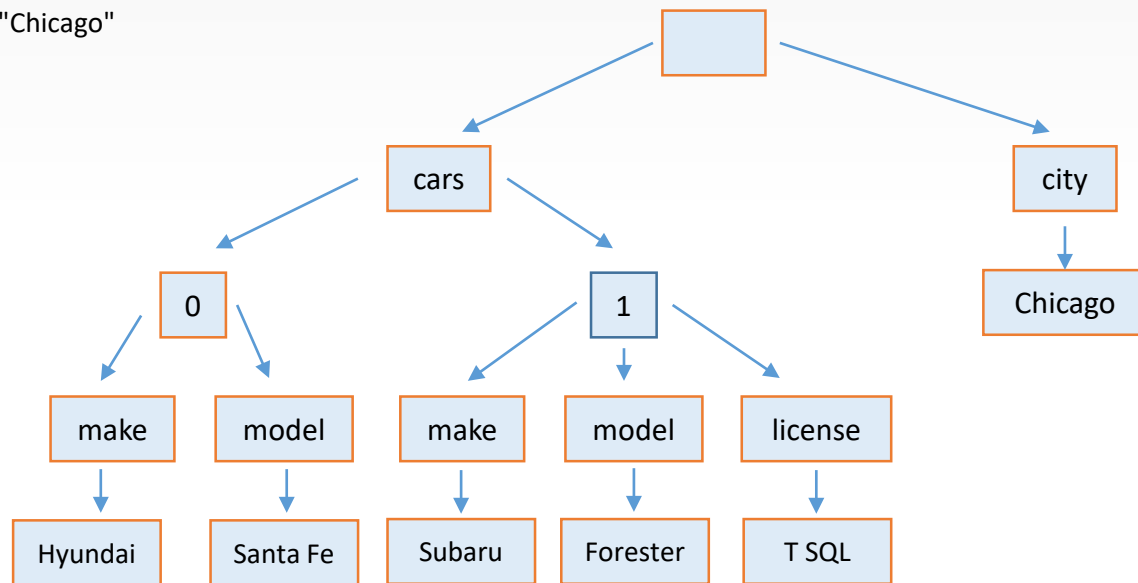
---

- Schema-agnostic
- Automatic
  - Every property of every record is indexed by default
  - No latches involved (remember it's highly write-optimized)
- Customizable
  - You can define what is indexed (and save space)



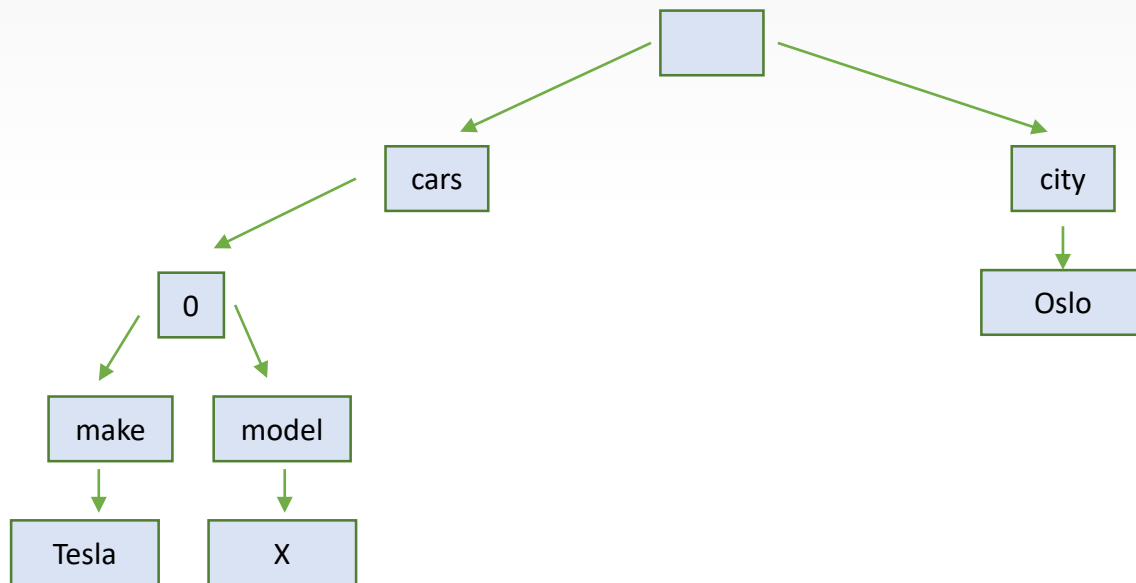
# Indexing

```
{ "cars": [  
  { "make": "Hyundai", "model": "Santa Fe" },  
  { "make": "Subaru", "model": "Forester", "plate": "T SQL" }  
],  
  "city": "Chicago"  
}
```

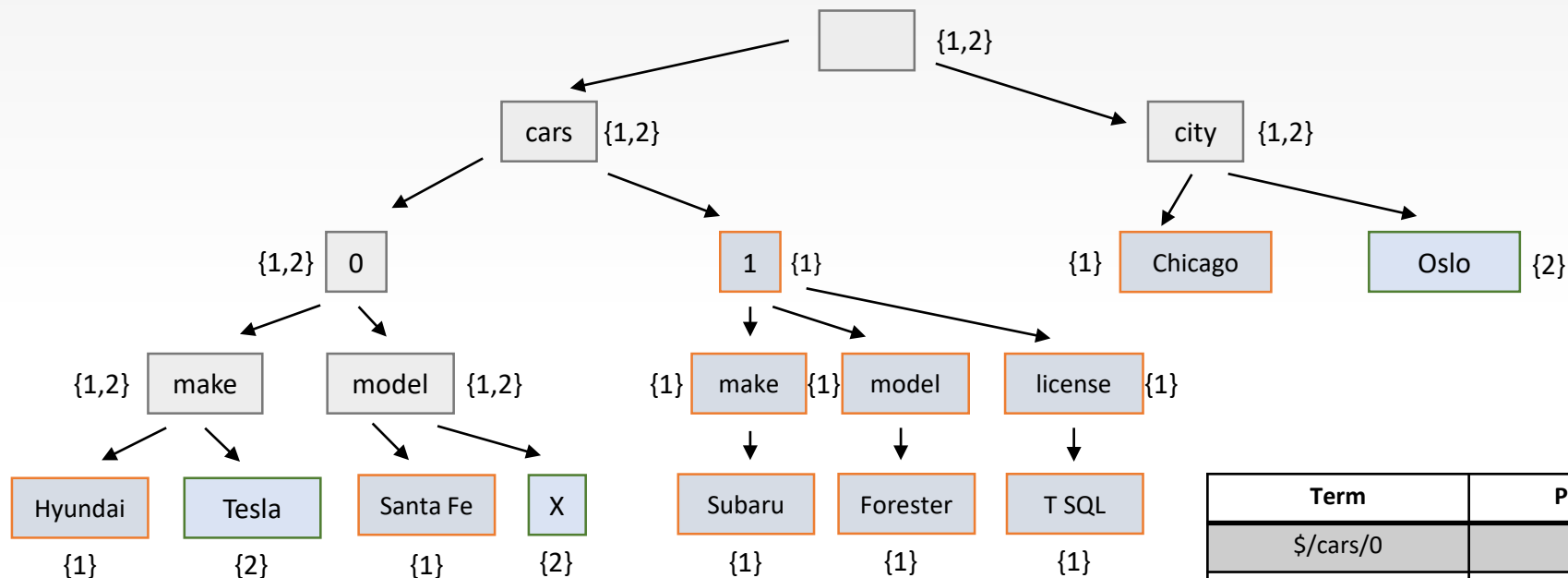


# Indexing

```
{ "cars": [  
  { "make": "Tesla", "model": "X" }  
],  
  "city": "Oslo"  
}
```



# Indexing



Term	Postings
\$/cars/0	1,2
\$/cars/0/make	1,2
\$/cars/0/model	1,2
\$/cars/1	1
.....	



# Indexing

## Indexing Policy

```
1 {  
2   "indexingMode": "consistent",  
3   "automatic": true,  
4   "includedPaths": [  
5     {  
6       "path": "/*",  
7       "indexes": [  
8         {  
9           "kind": "Range",  
10          "dataType": "Number",  
11          "precision": -1  
12        },  
13        {  
14          "kind": "Range",  
15          "dataType": "String",  
16          "precision": -1  
17        },  
18        {  
19          "kind": "Spatial",  
20          "dataType": "Point"  
21        }  
22      ]  
23    }  
24  ]  
25 }
```

# Backups



# Backups

---

- Remember what I said about indexes?



# Backups

---

- Backups are automatic
- Snapshots taken and stored separately in Azure Blob Storage
- For speed, it's written to same region as current Cosmos DB write region
- For safety, it's replicated to another region as well



# Backups

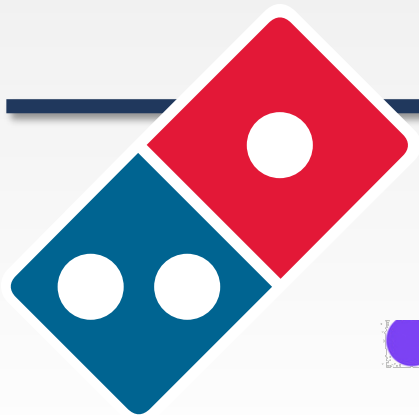
---

- Taken every 4 hours
- Only the last 2 snapshots are retained
- *“If the data is accidentally dropped or corrupted, contact Azure support within eight hours.”*
- You can maintain your own backups
  - Azure Cosmos DB Data Migration Tool “export to JSON” option
- If you delete a container/database, backups retained for 30 days



# Who's Using It?

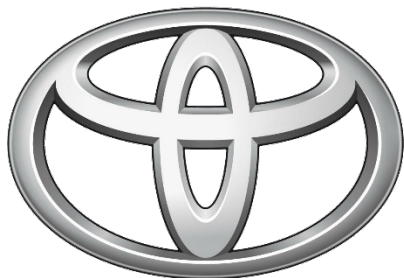




**CITRIX<sup>®</sup>**

**jet**

**vpps**



**Microsoft**

**TOYOTA**

<https://docs.microsoft.com/en-us/azure/cosmos-db/use-cases>



**HERAFLUX**  
TECHNOLOGIES<sup>®</sup>



# What's It Cost?



# Cosmos DB Pricing



- You Pay For:
  - Storage £0.1864 per GB/month
  - Throughput (single region writes): £4.3527 per 100 RU/sec
  - Throughput (multiple region writes): £8.7053 per 100 RU/sec
  - Data Transfer for geo-replication (varies by region)
    - UK South: £0.065per GB
  - Reserved Capacity can offer additional savings
- \*\*\**Check Azure Portal for most current pricing info*
- <https://azure.microsoft.com/en-us/pricing/details/cosmos-db/>



# Wanna Play For Free?

---



- There's a Cosmos DB Emulator!
- Run locally on your machine for free
- <https://docs.microsoft.com/en-us/azure/cosmos-db/local-emulator>



Azure Cosmos DB Emul. x

+ v

← → ↺ 🏠

🔒 https://localhost:8081/\_explorer/index.html

Azure Cosmos DB Emulator

Quickstart

Explorer

Feedback

New Collection

New SQL Query

New Stored Procedure v

↑ Upload

Delete Collection

Delete Database

⚙ Settings

SQL API

🔄 <

Documents x

New Document

💾 Update

↶ Discard

🗑 Delete

SELECT \* FROM c

Edit Filter

id

/Inventory...

🔄

6919167
6919168
6913043
2845513
6913186
6913205
6913200
2847659
6910038
6913203

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

```
"Tow Date": "2018-03-30T00:00:00.000000Z",
"Make": "LEXS",
"Style": "LL",
"Model": null,
"Color": "TAN",
"Plate": 3875313,
"State": "IN",
"Towed to Address": "701 N. Sacramento",
"Tow Facility Phone": "(773) 265-7605",
"id": "6919168",
"_rid": "zmNjAJbGEAACAAAAAAAAA==",
"_self": "dbs/zmNjAA=/colls/zmNjAJbGEAA=/docs/zmNjAJbGEAACAAAAAAAAA==/",
"_etag": "\"00000000-0000-0000-2f85-95e9221001d4\"",
"attachments": "attachments/",
"_ts": 1533780389
```

# Wanna Play For Free in the Cloud?



- Azure Cosmos DB 30 day Trial
- You can renew this unlimited times!
- Can globally distribute it to up to 3 regions
- “Use any of the capabilities Azure Cosmos DB provides for 30 days”
- After that, you can renew and load your data again
- <https://azure.microsoft.com/en-us/blog/try-azure-cosmosdb-for-30-days-free-no-commitment-or-sign-up/>





# FEEDBACK FORMS

PLEASE FILL OUT AND PASS TO YOUR ROOM  
HELPER BEFORE YOU LEAVE THE SESSION

# Questions?



@sqlbob



bob@bobpusateri.com



bobpusateri.com



bobpusateri

## THANK YOU FOR ATTENDING!

