

Azure Cosmos DB Overview



Reza Salehi

CLOUD CONSULTANT

@zaalion [linkedin.com/in/rezasalehi2008](https://www.linkedin.com/in/rezasalehi2008)



Overview



Understanding common Cosmos DB concepts

- Global distribution & multi-homing
- Consistency levels
- Auto-expire using time-to-live (TTL)
- Data partitioning & best practices

Cosmos DB APIs

- Which API is right for you?

Secure your Cosmos DB instance

- RBAC, keys, firewall & CORS

Summary



“Azure Cosmos DB is Microsoft's globally distributed, multi-model database service.”

Microsoft



Cosmos DB Concepts

**Global distribution
and multi-homing**

Data consistency levels

Time-to-live (TTL)

Data partitioning



Global Distribution & Multi-homing



Global Distribution

Click on a location to add or remove regions from your Azure Cosmos DB account.

* Each region is billable based on the throughput and storage for the account. [Learn more](#)



Configure regions

Enable multi-region writes ⓘ

[Disable](#) [Enable](#)

Configure the regions for reads, writes and availability zone (supported in selected regions and can only be configured when a new region is added). [+ Add region](#)

REGIONS	READS ENAB...	WRITES ENA...	AVAILABILIT...	ACTION
East US	✓	✓		🗑️
West US	✓	✓		🗑️



Global Distribution



Cosmos DB allows you to add or remove any of the Azure regions to your Cosmos account at any time with a click of a button



Choosing the required regions depends on the global reach of your application and where your users are located



Cosmos DB will seamlessly replicate your data to all the regions associated with your Cosmos account



The multi-homing capability of Cosmos DB, allows your application to be highly available



Multi-homing APIs

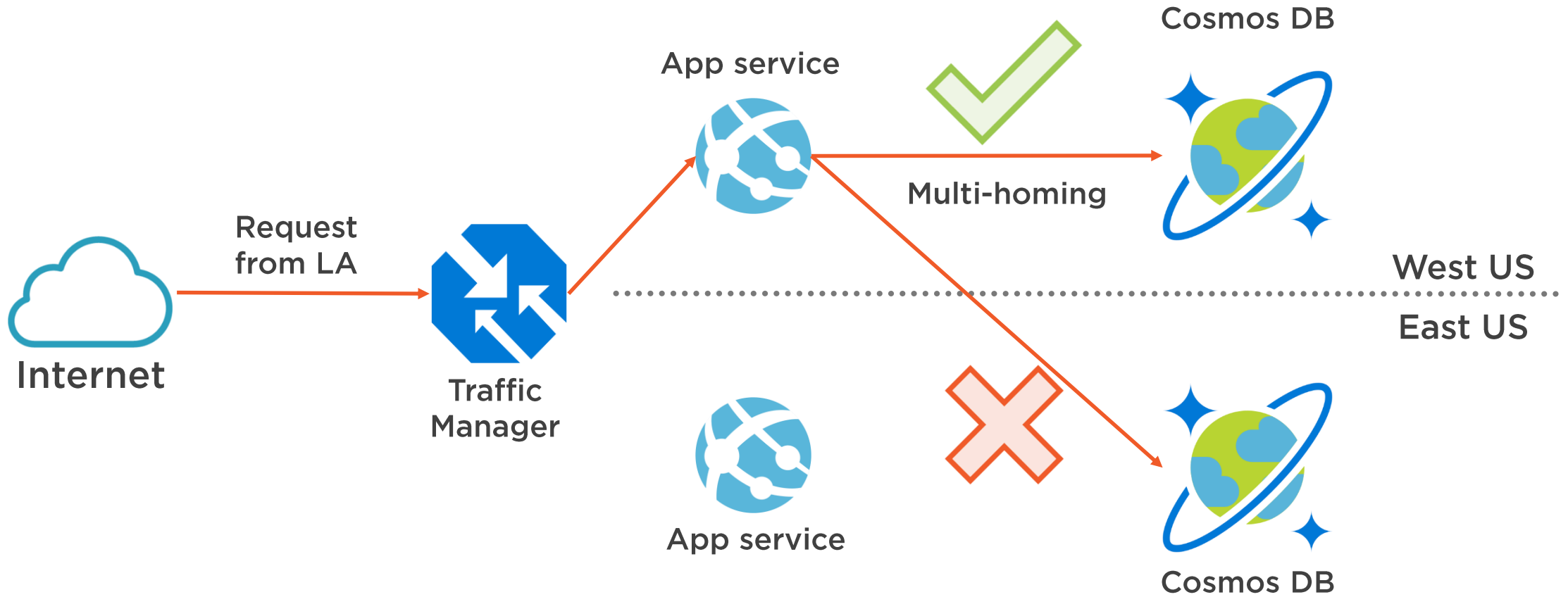
Your application is aware of the nearest region and can send requests to that region

Nearest region is identified without any configuration changes

When a new region is added or removed, the connection string stays the same



Multi-homing APIs



Time-to-live (TTL) & Consistency Levels



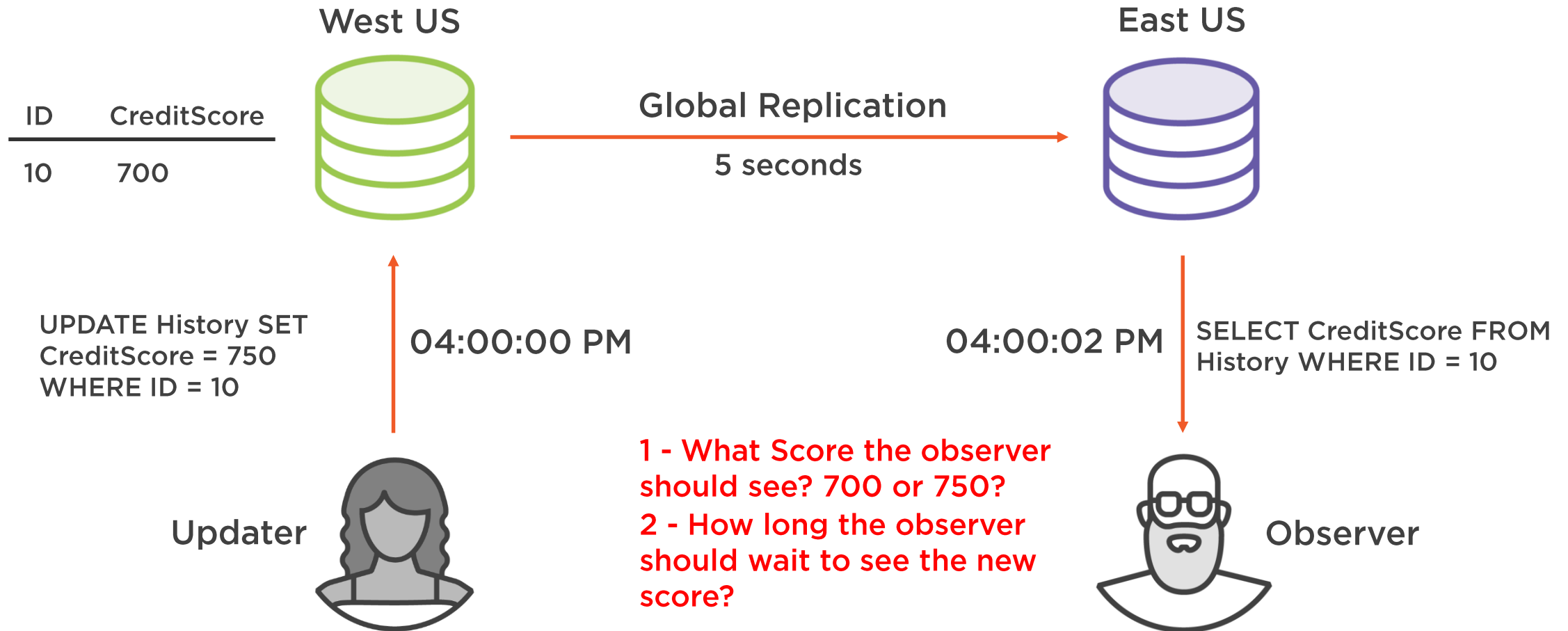
Time-to-live (TTL)

You can set the expiry time on Cosmos DB data items; the setting is called TTL and is set as seconds

Cosmos DB will automatically remove the items after this time period, since the last modified time



Data Consistency

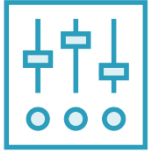


1 - What Score the observer should see? 700 or 750?
2 - How long the observer should wait to see the new score?

Depends on the scenario;
Social media comment or
banking.



Cosmos DB Consistency Levels



Data consistency versus data availability & latency



Most distributed databases ask you to choose between the two extreme consistency levels: strong and eventual consistency



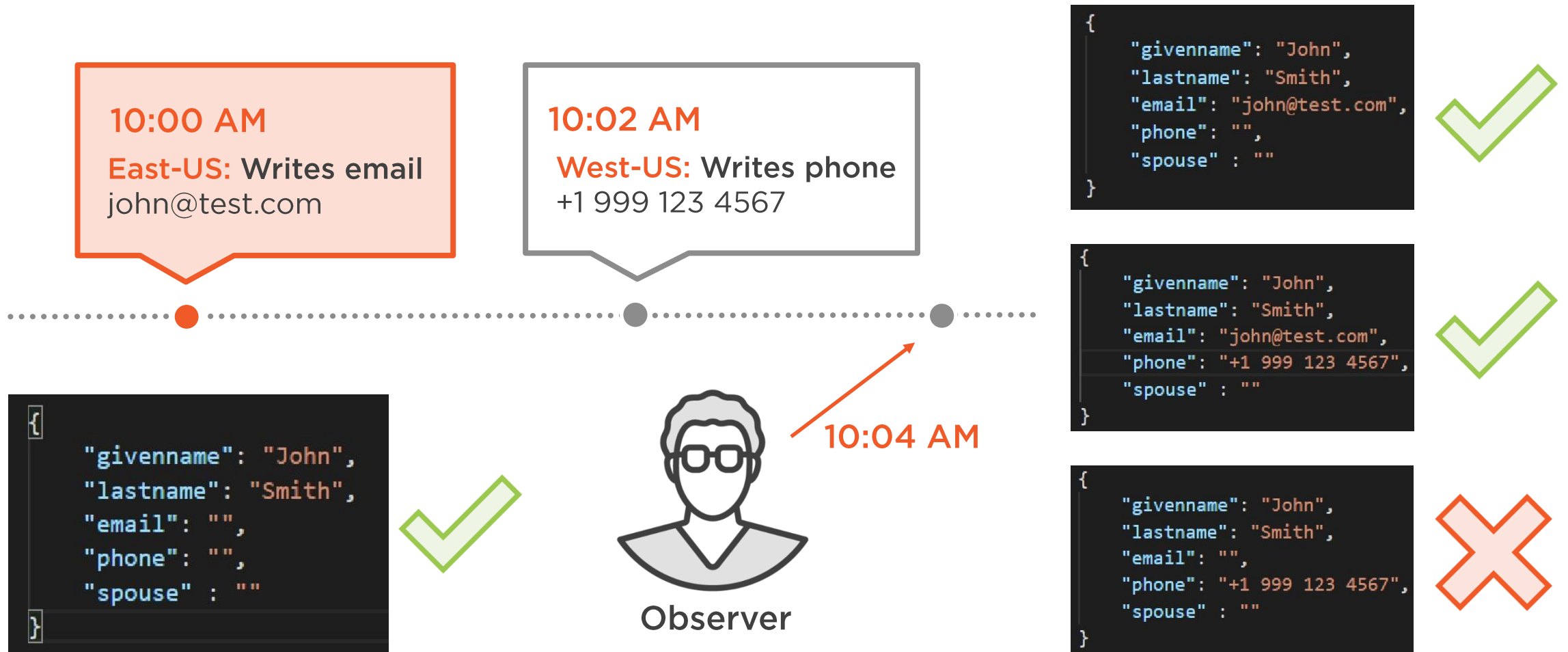
Azure Cosmos DB offers 5 consistency levels to choose from, including the 2 extremes



Strong, bounded staleness, session, consistent prefix, eventual



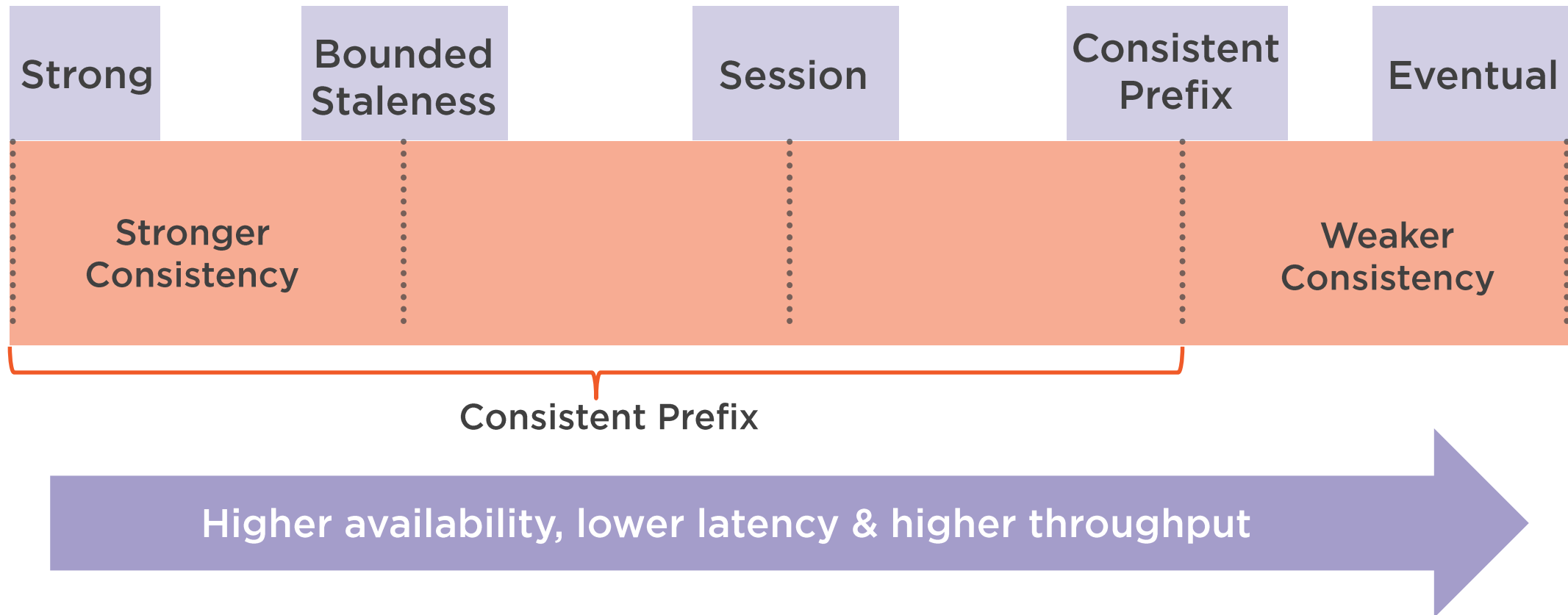
Multiple Writes and Consistent Prefix



Consistent prefix
consistency level
guarantees that reads
never see out-of-order
writes.



Cosmos DB Consistency Levels



Partitioning



Databases and Containers

**An Azure Cosmos database is
a unit of management for a
set of containers**

**A database consists of a set of
schema-agnostic containers**



Data Partitioning

A logical partition consists of a set of items that have the same partition key

Data that's added to the container is automatically partitioned across a set of logical partitions

Logical partitions are mapped to physical partitions that are distributed among several machines

Throughput provisioned for a container is divided evenly among physical partitions



Data Partitioning Best Practices



Pick a partition key that doesn't result in "hot spots" within your applications



Choose a partition key that has a wide range of values, so your data is evenly spread across logical partitions



For partition keys, use properties that appear frequently as a filter in queries; including the partition key in the filter predicate improves query performance



A single logical partition has a limit of 10 GB of storage



You can batch update items within a logical partition by using a transaction.



Cosmos DB APIs & Security



Cosmos DB APIs

SQL API

Recommended for
new applications

MongoDB API

Easy migration of
existing apps

Table API

Easy migration from
table storage

Cassandra API

Easy migration of
existing apps

Gremlin API

Both new and existing
apps



Security

Use RBAC to grant permission to users, groups or applications

Use firewall to limit clients who can access Cosmos DB

Deploy Cosmos DB into a VNet & use NSGs

CORS

Read-only and read-write keys




RBAC

[Check access](#) [Role assignments](#) [Deny assignments](#) [Classic administrators](#) [Roles](#)


Check access

Review the level of access a user, group, service principal, or managed identity has to this resource. [Learn more](#)

Find 

Azure AD user, group, or service principal


Search by name or email address



Add a role assignment

Grant access to resources at this scope by assigning a role to a user, group, service principal, or managed identity.


Add [Learn more](#)



View role assignments

View the users, groups, service principals and managed identities that have role assignments granting them access at this scope.

View [Learn more](#)



View deny assignments

View the users, groups, service principals and managed identities that have been denied access to specific actions at this scope.

[View](#)



VNet & Firewall

Allow access from

☐ All networks ☒ Selected networks

Configure network security for your Azure Cosmos DB account. [Learn more.](#)

Virtual networks

Secure your Azure Cosmos DB account with virtual networks. [+ Add existing virtual network](#) [+ Add new virtual network](#)

VIRTUAL NETWORK	SUBNET	ADDRESS RANGE	ENDPOINT STATUS	RESOURCE GROUP	SUBSCRIPTION
No network selected.					

Firewall

Add IP ranges to allow access from the internet or your on-premises networks. [+ Add my current IP \(72.143.192.89\)](#) ⓘ

IP (SINGLE IPV4 OR CIDR RANGE)

No IP range filter configured.

Exceptions

☐ Accept connections from within public Azure datacenters ⓘ

☒ Allow access from Azure Portal ⓘ



Read-only and Read-write Keys

Read-write Keys

Read-only Keys

URI

https://ps-demo.documents.azure.com:443/

PRIMARY KEY

yEdX9I52RO3Rg5QC8v7r25GBephF9LFgCBM5SSQu7FOuq5W5JTCxRzDtYCjY3UHVOERDZZ21vWlhdcSY8Ggtng==

SECONDARY KEY

xAjqTgtKXAicHq3SHzUxWFYgpNoiiksFWvaHoebwllsqDaKlq7PGVhwwq5LYouBic4CPcQjsiWPL25oTefRgbQ==

PRIMARY CONNECTION STRING

AccountEndpoint=https://ps-demo.documents.azure.com:443/;AccountKey=yEdX9I52RO3Rg5QC8v7r25GBephF9LFgCBM5SSQu7FOuq5W5JTCxRzDtYCjY3UHVOERDZZ21vWlh...

SECONDARY CONNECTION STRING

AccountEndpoint=https://ps-demo.documents.azure.com:443/;AccountKey=xAjqTgtKXAicHq3SHzUxWFYgpNoiiksFWvaHoebwllsqDaKlq7PGVhwwq5LYouBic4CPcQjsiWPL25oT...



CORS

 Submit  Discard

Cross-Origin Resource Sharing (CORS) is an HTTP feature that enables a web application running under one domain to access resources in another domain. Web browsers implement a security restriction known as same-origin policy that prevents a web page from calling APIs in a different domain. CORS provides a secure way to allow one domain (the origin domain) to call APIs in another domain.

Once you set the CORS rules for your Azure Cosmos account, a properly authenticated request made to the service from a browser client will be evaluated to determine whether it is allowed according to the rules you have specified.

Specify a comma-separated list of origins that will be allowed to make cross-origin calls to your Cosmos DB account.

Allowed Origins

For example: `https://domain-a, https://domain-b:12345`



Demo



Provisioning a new Cosmos DB instance
Containers, partitions, & TTL



Demo



Configuring Cosmos DB security



Summary



Common Cosmos DB concepts

- Global distribution & multi-homing
- Consistency levels
- Time-to-live (TTL)
- Data partitioning best practices

Which Cosmos DB API is right for you?

Secure your Cosmos DB instance

- RBAC, keys, firewall & CORS

