

Visual Studio **LIVE!** EXPERT SOLUTIONS FOR .NET DEVELOPERS | Austin

June 16, 2022
8:00am – 10:45am

Level:
Intermediate

Cosmos DB Deep Dive

Leonard Lobel
Chief Technology Officer
Sleek Technologies

We're Gonna Code Like It's 2018! 

Visual Studio 25 YEARS OF CODING INNOVATION

About Me

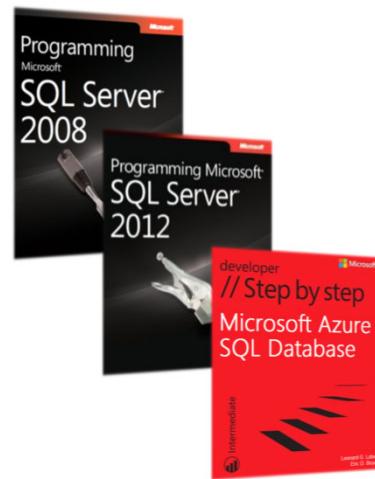


Leonard Lobel

- CTO & Co-Founder
 - Sleek Technologies, Inc.
- Principal Consultant
 - Tallan, Inc.
- Microsoft MVP
 - Data Platform
- Trainer/Speaker/Author
- Programming since 1979

Contact

- Email: lenni.lobel@sleektech.com
- Blog: lennilobel.wordpress.com
- Twitter: [@lennilobel](https://twitter.com/lennilobel)



Download Slides and Code

[http://bit.ly/
vslaustin2022_cosmosdb](http://bit.ly/vslaustin2022_cosmosdb)

(all lower case!)

Agenda

Part I

Introduction to
Azure Cosmos DB

- Overview
- Management
- Indexing
- Multiple APIs
- Migration
- Throughput
- Partitioning
- Global distribution

Part II

Building Event-driven Microservices
with the Cosmos DB Change Feed

- Replication
- Denormalization
- Notifications
- Materialized views
- Data movement

Agenda

Part I

Introduction to Azure Cosmos DB

- Overview
- Management
- Indexing
- Multiple APIs
- Migration
- Throughput
- Partitioning
- Global distribution

Part II

Building Event-driven Microservices with the Cosmos DB Change Feed

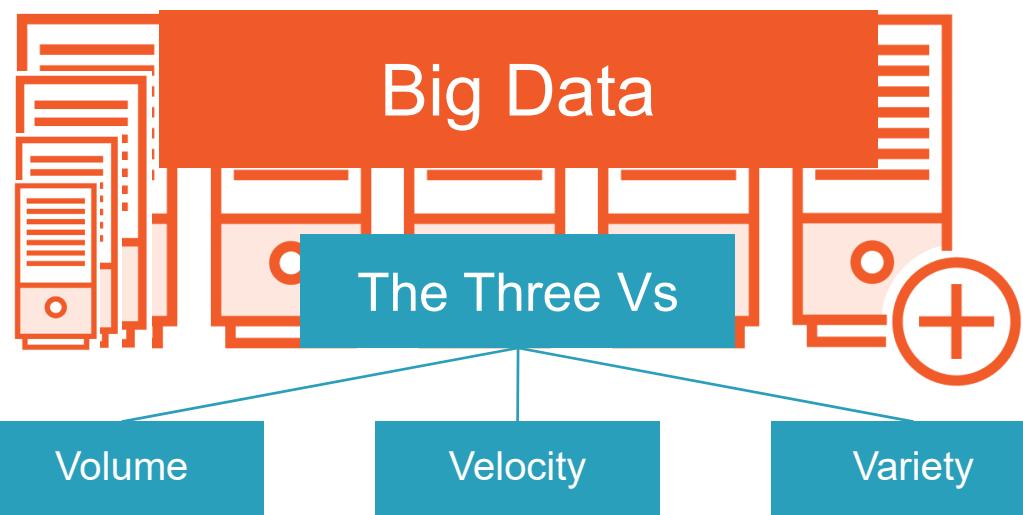
- Replication
- Denormalization
- Notifications
- Materialized views
- Data movement

Introduction to Azure Cosmos DB

Introduction to Azure Cosmos DB

Overview

What is NoSQL?



What is a NoSQL database?

Distributed

Replicas ensure high availability and resilience



Scale-out

Horizontal partitioning for elastic storage and throughput



Schema-free

No enforced schema
No defined shape



What is Cosmos DB?

Massively scalable NoSQL database

Fully managed Azure PaaS
99.999% SLAs
Single-digit millisecond reads and writes



Four Nines (99.99%)

Automatic Horizontal Partitioning
Max 5256 partition downtime per year



Five Nines (99.999%)

Elastic scaling of storage and throughput
Max 326 minutes downtime per year

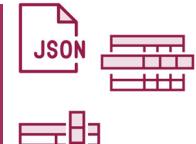
Global distribution

Point-and-click geo-replication
Multiple write regions



Multi-model / Multi-API

Not exclusively a document database
Also supports table, graph, and columnar



Getting Started

The screenshot shows a web browser window titled "Try Azure Cosmos DB | Microsoft". The main content area is divided into three sections:

- 30-day Free Trial**
<http://azure.microsoft.com/try/cosmosdb>
- Azure Portal**
<http://portal.azure.com>
- Azure Subscription**
<http://azure.microsoft.com>

Below these sections, it says "Free Tier" and "First ~\$64/mo free". To the right, there's a large blue graphic featuring two white clouds and a blue ring.

At the bottom, a note states: "With Azure Cosmos DB free tier, you will get the first 1000 RU/s and 25 GB of storage for free in an account. You can enable free tier on up to one account per subscription. Estimated \$64/month discount per account." There are two radio buttons: Apply and Do Not Apply.

Getting Started

The screenshot shows the same landing page as above, but with three additional callout boxes:

- 30-day Free Trial**
<http://azure.microsoft.com/try/cosmosdb>
- Azure Portal**
<http://portal.azure.com>
- Azure Subscription**
<http://azure.microsoft.com>

Below these, it says "Free Tier" and "First ~\$64/mo free". To the right, there's a large blue graphic featuring two white clouds and a blue ring.

Local Emulator
<http://aka.ms/cosmosdb-emulator>
No internet connection required

In the bottom right corner of the slide, there is a small circular icon with a double-headed arrow symbol.

Demo



Creating a Cosmos DB account



Microsoft AzureSearch resources, services, and docs (G+/-)DashboardRefreshFull screenEditShareDownloadCloneAssign tagsDeleteFeedback

Last updated: a few seconds ago

Quickstarts + tutorials	
 Windows Virtual Machines	Provision Windows Server, SQL Server, SharePoint VMs
 Linux Virtual Machines	Provision Ubuntu, Red Hat, CentOS, SUSE, CoreOS VMs
 App Service	Create Web Apps using .NET, Java, Node.js, Python, PHP
 Functions	Process events with a serverless code architecture
 SQL Database	Managed relational SQL Database as a Service

The screenshot shows the Azure portal interface. On the left, there's a sidebar with various service icons and links like Home, Dashboard, All services, and Favorites. A red box highlights the 'Create a resource' button at the top of the sidebar. The main content area has a search bar at the top. Below it are several cards representing different Azure services: Virtual machines, App Services, Cloud services (classic), SQL databases, Subscriptions, Monitor, Security Center, Advisor, Help + support, and Cost Management + Billing. A message at the bottom right says 'Last updated: a minute ago'.

The screenshot shows the Microsoft Azure Marketplace 'New' section. It features a search bar at the top and a 'Popular' section with several items: Windows Server 2016 Datacenter, Ubuntu Server 18.04 LTS, Web App, SQL Database, Function App, and Azure Cosmos DB. The Azure Cosmos DB item is highlighted with a red box. To the left, there's a sidebar with categories like Get started, Recently created, AI + Machine Learning, Analytics, Blockchain, Compute, Containers, Databases, Developer Tools, DevOps, Identity, Integration, Internet of Things, IT & Management Tools, Media, and Migration. A 'Popular' tab is also visible.

The screenshot shows the Microsoft Azure portal interface for creating a new Azure Cosmos DB account. The top navigation bar includes the Microsoft Azure logo, a search bar, and user information for 'lenni@hotmail.com LENNI LOBEL'. Below the header, the breadcrumb navigation shows 'Dashboard > Create a resource > Create Azure Cosmos DB Account'. The main title is 'Create Azure Cosmos DB Account'. The 'Basics' tab is selected, while other tabs like 'Global Distribution', 'Networking', 'Backup Policy', 'Encryption', 'Tags', and 'Review + create' are visible. A note at the top states: 'Azure Cosmos DB is a fully managed NoSQL database service for building scalable, high performance applications. Try it for free, for 30 days with unlimited renewals. Go to production starting at \$24/month per database, multiple containers included.' A link to 'Learn more' is provided. The 'Project Details' section asks to select a subscription and resource group. A red box highlights the 'Subscription' dropdown set to 'Microsoft Azure Sponsorship' and the 'Resource Group' dropdown where 'cosmos-demos-rg' is typed and highlighted. The 'Instance Details' section includes fields for 'Account Name' (placeholder 'Enter account name'), 'API' (set to 'Core (SQL)'), 'Location' (set to '(US) West US'), and 'Capacity mode' (radio button selected for 'Provisioned throughput'). Below these fields is a link to 'Learn more about capacity mode'. At the bottom of the screen are buttons for 'Review + create' (highlighted in blue), 'Previous', and 'Next: Global Distribution'.

This screenshot shows the same Azure portal interface for creating an Azure Cosmos DB account, but with a different configuration. The 'Resource Group' dropdown now shows 'cosmos-demos-rg' instead of 'Create new'. The rest of the form, including the 'Project Details' and 'Instance Details' sections, remains identical to the first screenshot. The 'Review + create' button is still highlighted in blue at the bottom.

The screenshot shows the 'Create Azure Cosmos DB Account' wizard on the 'Basics' step. The account name 'lennidemo' is highlighted with a red box. Other fields include 'Subscription' (Microsoft Azure Sponsorship), 'Resource Group' (cosmos-demos-rg), 'API' (Core (SQL)), 'Location' ((US) West US), and 'Capacity mode' (Provisioned throughput selected). Navigation buttons at the bottom are 'Review + create', 'Previous', and 'Next: Global Distribution'.

This screenshot is identical to the one above, but the 'API' dropdown has been changed to 'No API' (MongoDB, Gremlin, or Cassandra). The rest of the configuration remains the same, with the account name 'lennidemo' still highlighted in red.

The screenshot shows the 'Create Azure Cosmos DB Account' wizard on the 'Instance Details' step. The 'Subscription' dropdown is set to 'Microsoft Azure Sponsorship'. The 'Resource Group' dropdown is set to 'cosmos-demos-rg'. The 'Account Name' field contains 'lennidemo'. The 'API' dropdown is currently empty and highlighted with a red box. The 'Location' is '(US) West US'. The 'Capacity mode' is set to 'Provisioned throughput'. The 'Apply Free Tier Discount' button is selected. At the bottom, there are 'Review + create' and 'Next: Global Distribution' buttons.

The screenshot shows the 'Create Azure Cosmos DB Account' wizard on the 'Global Distribution' step. The tabs at the top are Basics, Global Distribution (selected), Networking, Backup Policy, Encryption, Tags, and Review + create. Under 'Global Distribution', it says 'Configure global distribution and regional settings for your account. You can also change these settings after the account is created.' Two sections are highlighted with red boxes: 'Geo-Redundancy' with 'Enable' selected and 'Multi-region Writes' with 'Disable' selected. At the bottom, there are 'Review + create' and 'Next: Networking' buttons.

The screenshot shows the 'Create Azure Cosmos DB Account' wizard in the Microsoft Azure portal. At the top, there's a green validation message: 'Validation Success'. Below it, tabs for 'Basics', 'Global Distribution', 'Networking', 'Backup Policy', 'Encryption', 'Tags', and 'Review + create' are visible, with 'Review + create' being the active tab. A red box highlights the 'Creation Time' section, which displays 'Estimated Account Creation Time (in minutes)' set to 15. A tooltip below the input field states: 'The estimated creation time is calculated based on the location you have selected'. The 'Basics' section shows the following configuration:

Subscription	Microsoft Azure Sponsorship
Resource Group	cosmos-demos-rg
Location	West US
Account Name	(new) lennidemo
API	Core (SQL)
Capacity mode	Provisioned throughput
Geo-Redundancy	Disable
Multi-region Writes	Disable

At the bottom, a large blue 'Create' button is highlighted with a red box, and navigation buttons for 'Previous', 'Next', and 'Download a template for automation' are present.

The screenshot shows the 'Overview' page for the newly created Cosmos DB account, 'Microsoft.Azure.CosmosDB-20210522144809'. The left sidebar includes 'Deployment', 'Overview' (which is selected), 'Inputs', 'Outputs', and 'Template'. The main area displays deployment status: 'Deployment is in progress'. Deployment details show the name, subscription, and resource group. A table lists the resources and their status:

Resource	Type	Status	Operation details
lennidemo	Microsoft.DocumentDb/d...	OK	Operation details

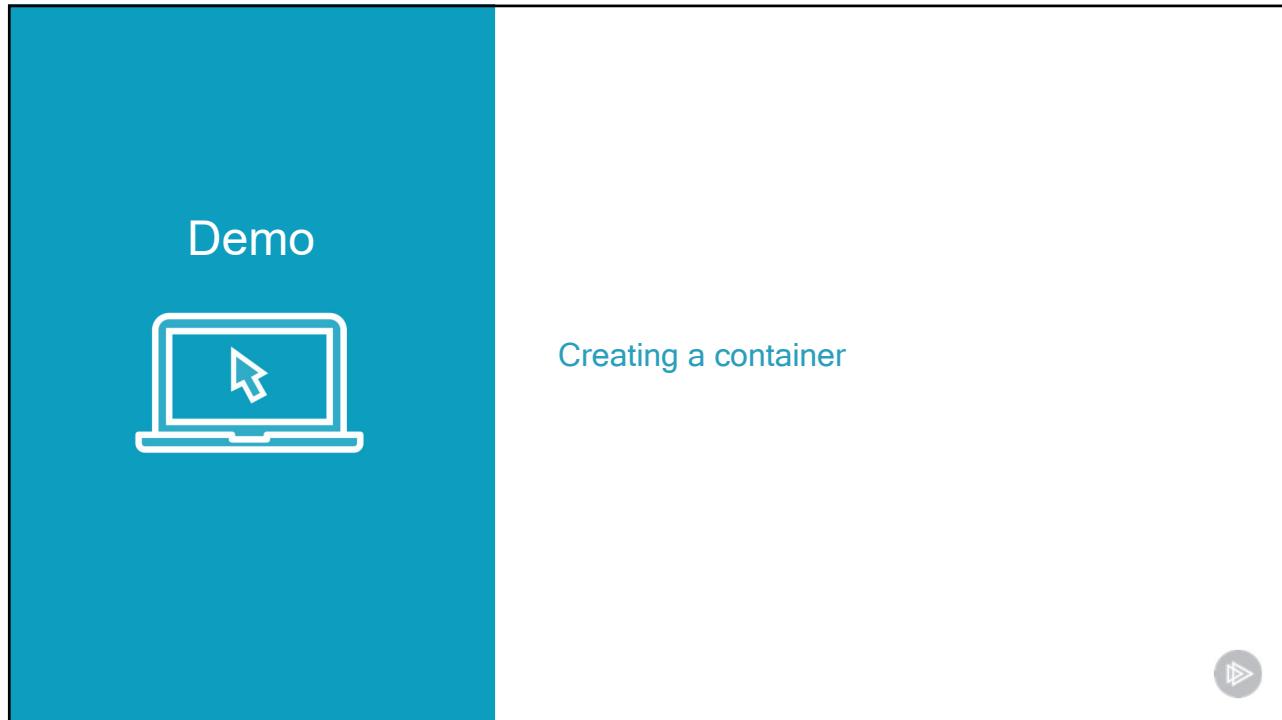
On the right side, there are links to 'Security Center', 'Free Microsoft tutorials', 'Work with an expert', and a 'Find an Azure expert' button. A feedback survey bar at the top right says: 'We'd love your feedback!'.

The screenshot shows the Microsoft Azure Deployment Overview page for a deployment named "Microsoft.Azure.CosmosDB-20210522144809". The main message is "Your deployment is complete". Deployment details include a deployment name, subscription, and resource group. A "Go to resource" button is highlighted with a red box. The right sidebar contains links to Security Center, Microsoft tutorials, and expert support.

The screenshot shows the Microsoft Azure Cosmos DB account overview for "lennidemo". The "Status" is listed as "Online". The "Keys" section is highlighted with a red box. The "Containers" section shows a message about having no containers yet. The "Monitoring" section allows setting data retention periods. The "Requests" section displays performance metrics.

This screenshot shows the 'Keys' section of the Azure portal for an Azure Cosmos DB account named 'lennidemo'. The left sidebar includes options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, Data Explorer (which is selected and highlighted with a red box), Settings, Features, Replicate data globally, Default consistency, Backup & Restore, Firewall and virtual networks, Private Endpoint Connections, CORS, and Keys. The main content area displays four connection keys: URI (https://lennidemo.documents.azure.com:443/), PRIMARY KEY (QUVal4Aw6DALPL4scS8lFKU2BW4lKhgjkj2j54KxZn5JL5HyMWm0mgqj8LbkRbpZtnm7kG8mNfYcfG0b5p1hRg==), SECONDARY KEY (LgBp1mEks9RyEZ3VflMnnlH3ra0ZZmbHlHA4fQD1aKrZp4XYHEjj0UMJMPTqiHDfDwYkz9Cgr0ixbd3Mi7g==), PRIMARY CONNECTION STRING (AccountEndpoint=https://lennidemo.documents.azure.com:443/;AccountKey=QUVal4Aw6DALPL4scS8lFKU2BW4lKhgjkj2j54KxZn5JL5HyMWm0mgqj8LbkRb...), and SECONDARY CONNECTION STRING (AccountEndpoint=https://lennidemo.documents.azure.com:443/;AccountKey=LgBp1mEks9RyEZ3VflMnnlH3ra0ZZmbHlHA4fQD1aKrZp4XYHEjj0UMJMPTqi...).

This screenshot shows the 'Data Explorer' section of the Azure portal for the same 'lennidemo' Azure Cosmos DB account. The left sidebar is identical to the previous screenshot. The main content area features a 'Welcome to Cosmos DB' message: 'Globally distributed, multi-model database service for any scale'. It includes two main buttons: 'Start with Sample' (with a planet icon) and 'New Container' (with a document icon). Below these are sections for 'Common Tasks' (New Database), 'Recents', and 'Tips' (Data Modeling, Cost & Throughput Calculation, Configure automatic failover). The bottom of the screen shows standard navigation icons.



Microsoft Azure lenni@hotmail.com LENNI LOBEL

Dashboard > lennidemo

lennidemo | Data Explorer ...

Azure Cosmos DB account

New Container

Enable Azure Synapse Link

Enable Notebooks (Preview)

SQL API

Search (Ctrl+/)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Notifications

Data Explorer

Settings

Features

Replicate data globally

Default consistency

Backup & Restore

Firewall and virtual networks

Private Endpoint Connections

CORS

Keys

New Container

Welcome to Cosmos DB

Globally distributed, multi-model database service for any scale

Start with Sample

New Container

Common Tasks

Recents

Tips

New Database

Data Modeling

Cost & Throughput Calculation

Configure automatic failover

The screenshot shows the Microsoft Azure Data Explorer interface for the 'lennidemo' account. The left sidebar has 'Data Explorer' selected. A modal dialog titled 'New Container' is open on the right. The 'Database id' field is highlighted with a red box and contains the value 'Families'. Other settings like throughput and autoscale are visible.

This screenshot is identical to the one above, but the 'Database id' field in the 'New Container' dialog now contains the value 'Families' instead of 'Families1'.

The screenshot shows the Microsoft Azure Data Explorer interface for the 'lennidemo' database. The left sidebar has 'Data Explorer' selected. A modal window titled 'New Container' is open on the right. The modal contains fields for 'Database id' (set to 'Create new'), 'Container id' (set to 'Families'), and 'Partition key' (set to '/address/zipCode'). The 'Share throughput across containers' checkbox is unchecked. A note at the top of the modal says: 'With free tier, you'll get the first 400 RU/s and 5 GB of storage in this account for free. Billing will apply if you provision more than 400 RU/s of manual throughput, or if the container scales beyond 400 RU/s with autoscale.' An 'OK' button is at the bottom right.

This screenshot is identical to the one above, but with different input values in the 'New Container' dialog. The 'Container id' field now contains 'Families' (with a red box around it), and the 'Partition key' field now contains '/address/zipCode' (also with a red box around it). All other settings remain the same.

The screenshot shows the Microsoft Azure Data Explorer interface for the 'lennidemo' database. On the left, the navigation menu is open, showing options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer (which is selected). The main area displays a 'Welcome to Cosmos DB' message and common tasks such as 'Start with Sample' and 'New Database'. A 'New Container' dialog is open on the right. It contains fields for 'Database id' (set to 'Create new' and 'Families'), 'Container id' (set to 'Families'), and 'Partition key' (set to '/address/zipCode'). The 'Partition key' field is highlighted with a red box.

This screenshot is similar to the one above, showing the 'New Container' dialog in the Azure Data Explorer. The 'Partition key' field is still highlighted with a red box. In this view, the 'Container throughput (autoscale)' section is also highlighted with a red box. It includes a radio button for 'Autoscale' (selected) or 'Manual', a note about estimating RU/s using a calculator, and a field for 'Container Max RU/s' set to '4000'. A note at the bottom states: 'Your container throughput will automatically scale from 400 RU/s (10% of max RU/s) - 4000 RU/s based on usage.'

The screenshot shows the Microsoft Azure Data Explorer interface for an account named 'lennidemo'. The left sidebar has 'Data Explorer' selected. A modal window titled 'New Container' is open on the right. Inside the modal, under the 'Container throughput' section, there is a field for 'Container throughput (400 - unlimited RU/s)' with a value of '400'. This entire section is highlighted with a red box. Below it, there is a note about estimated costs: '\$0.032 hourly / \$0.77 daily / \$23.36 monthly'. The 'OK' button at the bottom right of the modal is also highlighted with a red box.

This screenshot is similar to the one above, showing the 'New Container' dialog in the Microsoft Azure Data Explorer. The 'Analytical store' section is highlighted with a red box. It contains a radio button for 'On' and another for 'Off'. A note states: 'Azure Synapse Link is required for creating an analytical store container. Enable Synapse Link for this Cosmos DB account.' Below this is an 'Enable' button. The 'OK' button at the bottom right is also highlighted with a red box.

The screenshot shows the Microsoft Azure Data Explorer interface for the 'lennidemo' Azure Cosmos DB account. The left sidebar menu is visible, showing various options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, Data Explorer (which is selected and highlighted in grey), Settings, Features, Replicate data globally, Default consistency, Backup & Restore, Firewall and virtual networks, Private Endpoint Connections, CORS, and Keys. The main content area displays a 'Welcome to Cosmos DB' page with the sub-headline 'Globally distributed, multi-model database service for any scale'. It features three main sections: 'Common Tasks' (with 'Start with Sample' and 'New Container' buttons), 'Recents' (empty), and 'Tips' (Data Modeling, Cost & Throughput Calculation, Configure automatic failover). A red box highlights the 'Families' button under the 'SQL API' dropdown in the top navigation bar.

This screenshot is identical to the one above, showing the Microsoft Azure Data Explorer interface for the 'lennidemo' Azure Cosmos DB account. The left sidebar and main content area are the same. The difference is in the top navigation bar's 'SQL API' dropdown, where the 'Families' item is now expanded, showing a nested 'Families' item. A red box highlights this nested 'Families' item.

Microsoft Azure

Search resources, services, and docs (G+)

Dashboard > lennidemo

lennidemo | Data Explorer

Azure Cosmos DB account

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Notifications

Data Explorer

Settings

Features

Replicate data globally

Default consistency

Backup & Restore

Firewall and virtual networks

Private Endpoint Connections

CORS

Keys

SQL API

Families

- Items
- Scale & Settings
- Stored Procedures
- User Defined Functions
- Triggers

Welcome to Cosmos DB

Globally distributed, multi-model database service for any scale

Start with Sample

New Container

Common Tasks

- New SQL Query
- Open Query
- New Stored Procedure

Recents

Tips

- Data Modeling
- Cost & Throughput Calculation
- Configure automatic failover

Demo

Creating documents

Microsoft Azure

Search resources, services, and docs (G+)

Dashboard > lennidemo

lennidemo | Data Explorer

Azure Cosmos DB account

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Notifications

Data Explorer

Settings

Features

Replicate data globally

Default consistency

Backup & Restore

Firewall and virtual networks

Private Endpoint Connections

CORS

Keys

New Container

Enable Azure Synapse Link

Enable Notebooks (Preview)

New SQL Query

Open Query

SQL API

Families

Items

Scale & Settings

Stored Procedures

User Defined Functions

Triggers

Welcome to Cosmos DB

Globally distributed, multi-model database service for any scale

Start with Sample

Get started with a sample provided by Cosmos DB

New Container

Create a new container for storage and throughput

Common Tasks

New SQL Query

Open Query

New Stored Procedure

Recents

Tips

Data Modeling

Learn more about modeling

Cost & Throughput Calculation

Learn more about cost calculation

Configure automatic failover

Learn more about Cosmos DB high-availability

Microsoft Azure

Search resources, services, and docs (G+)

Dashboard > lennidemo

lennidemo | Data Explorer

Azure Cosmos DB account

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Notifications

Data Explorer

Settings

Features

Replicate data globally

Default consistency

Backup & Restore

Firewall and virtual networks

Private Endpoint Connections

CORS

Keys

New Item

Upload Item

SQL API

Families

Items

SELECT * FROM c

Edit Filter

	id	/ad...

Scale & Settings

Stored Procedures

User Defined Functions

Triggers

Load more

Create new or work with existing document(s).

This screenshot shows the Microsoft Azure Data Explorer interface for an Azure Cosmos DB account named 'lennidemo'. The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer (which is selected). The main area displays the SQL API interface for the 'Families' collection. A red box highlights the 'Items' section where a JSON document is shown:

```
1 {
2   "id": "replace_with_new_document_id"
3 }
```

This screenshot shows the Microsoft Azure Data Explorer interface for the same Azure Cosmos DB account. The left sidebar and main interface are identical to the previous screenshot. A red box highlights the 'Save' button in the top right corner of the interface. The JSON document shown in the 'Items' section is identical to the one in the previous screenshot.

```
1 {
2   "familyName": "Smith",
3   "address": {
4     "addressLine": "123 Main Street",
5     "city": "Chicago",
6     "state": "IL",
7     "zipCode": "60601"
8   },
9   "parents": [
10    "Peter",
11    "Alice"
12 ],
13   "kids": [
14    "Adam",
15    "Jacqueline",
16    "Joshua"
17 ]
18 }
```

The screenshot shows the Microsoft Azure Data Explorer interface for an Azure Cosmos DB account named 'lennidemo'. The left sidebar navigation bar includes links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer (which is selected). The main area displays the SQL API results for the 'Families' collection. A red box highlights the JSON document returned for the item with ID '98806...'. The JSON document represents a family with the following details:

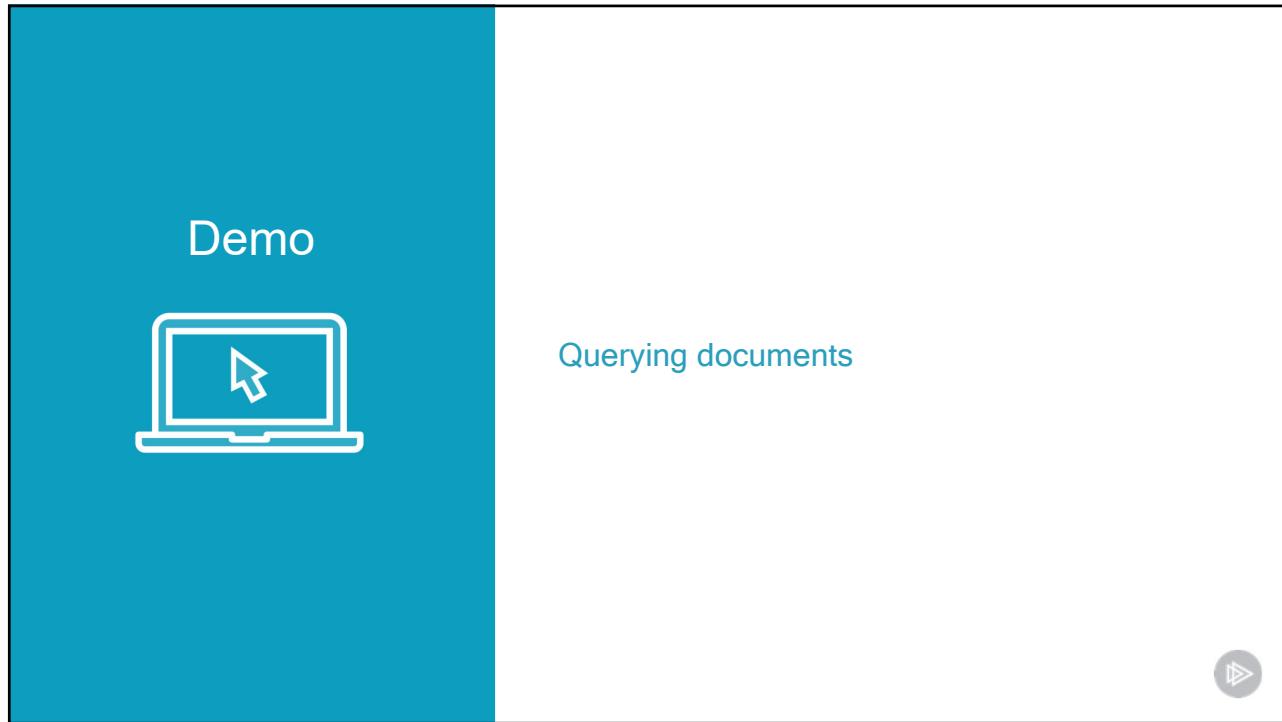
```

1  "familyName": "Smith",
2   "address": {
3     "addressLine": "123 Main Street",
4     "city": "Chicago",
5     "state": "IL",
6     "zipCode": "60601"
7   },
8   "parents": [
9     "Peter",
10    "Alice"
11 ],
12   "kids": [
13     "Adam",
14     "Jacqueline",
15     "Joshua"
16   ],
17   "id": "9880699f-2431-4620-9d45-e5b5bb890198",
18   "_rid": "n05WALB1pEkBAAAAAAA==",
19   "_self": " dbs/n05WALB1pEk=/colls/n05WALB1pEk=/docs/n05WALB1pEkBAAAAAAA==",
20   "_etag": "\\"7500ceba-0000-0700-0000-60a95a9f0000\\",
21   "_attachments": "attachments/",
22   "_ts": 1621711519
23
24
25
  
```

This screenshot is similar to the one above, showing the Microsoft Azure Data Explorer interface for the 'lennidemo' account. The left sidebar shows the 'Data Explorer' link is selected. The main area displays the SQL API results for the 'Families' collection. A red box highlights the JSON document returned for the item with ID '98806...'. The JSON document represents a family with the following details:

```

1  "familyName": "Jones",
2   "address": {
3     "addressLine": "456 Harbor Boulevard",
4     "city": "Chicago",
5     "state": "IL",
6     "zipCode": "60603"
7   },
8   "parents": [
9     "David",
10    "Diana"
11 ],
12   "kids": [
13     "Evan"
14   ],
15   "pets": [
16     "Lint"
17   ],
18   "id": "9880699f-2431-4620-9d45-e5b5bb890198",
19   "_rid": "n05WALB1pEkCAAAAAAA==",
20   "_self": " dbs/n05WALB1pEk=/colls/n05WALB1pEk=/docs/n05WALB1pEkCAAAAAAA==",
21   "_etag": "\\"7500dc3-0000-0700-0000-60a95b1e0000\\",
22   "_attachments": "attachments/",
23   "_ts": 1621711646
24
25
  
```



Dashboard > lennidemo

lennidemo | Data Explorer Azure Cosmos DB account

SQL API

New SQL Query

Items

SELECT * FROM c

Edit Filter

	id	/a...
1	98806...	60601
2	6e2a0...	60603
3	Load more	

```

1   "familyName": "Jones",
2   "address": {
3     "addressLine": "456 Harbor Boulevard",
4     "city": "Chicago",
5     "state": "IL",
6     "zipCode": "60603"
7   },
8   "parents": [
9     "David",
10    "Diana"
11   ],
12   "kids": [
13     "Evan"
14   ],
15   "pets": [
16     "Lint"
17   ],
18   "id": "6e2a075e-8709-444b-a06e-bf9a496978fb",
19   "_rid": "n05WALB1pEkCAAAAAAAA=",
20   "_self": "dbs/n05WALB1pEk=/colls/n05WALB1pEk=/docs/n05WALB1pEkCAAAAAA",
21   "_etag": "75008dc3-0000-0700-0000-60a95b1e0000",
22   "_attachments": "attachments/",
23   "_ts": 1621711646
24
25

```

This screenshot shows the Microsoft Azure Data Explorer interface for an Azure Cosmos DB account named 'lennidemo'. The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer (which is selected). The main area displays the SQL API interface. A tree view on the left shows 'Families' and 'Families' under it, with options for Items, Scale & Settings, Stored Procedures, User Defined Functions, and Triggers. The right panel has a search bar at the top and a 'Query 1' tab. The query editor contains the following SQL code:

```
1 SELECT * FROM c
```

Below the query editor, there is a message: "Execute a query to see the results".

This screenshot shows the same Microsoft Azure Data Explorer interface as the previous one, but with a WHERE clause added to the query. The 'Execute Query' button is highlighted with a red box. The query editor now contains:

```
1 SELECT * FROM c
2 WHERE c.address.city = 'Chicago'
```

The rest of the interface remains the same, including the tree view on the left and the message below the query editor.

This screenshot shows the Microsoft Azure Data Explorer interface for an Azure Cosmos DB account named "lennidemo". The left sidebar navigation bar is visible, with "Data Explorer" selected. In the main area, the "SQL API" tab is active, and the "Items" section under the "Families" collection is selected. A query titled "Query 1" is running, displaying the results of a SELECT statement:

```
1 SELECT * FROM c
2 WHERE c.address.city = 'Chicago'
```

The results pane shows two items, each representing a family record. One item is highlighted with a red box. The JSON output for the highlighted item is:

```
{
  "familyName": "Smith",
  "address": {
    "addressLine": "123 Main Street",
    "city": "Chicago",
    "state": "IL",
    "zipCode": "60601"
  },
  "parents": [
    "Peter",
    "Alice"
  ],
  "kids": [
    "Adam",
    "Jacqueline",
    "Joshua"
  ]
}
```

This screenshot shows the Microsoft Azure Data Explorer interface for the same Azure Cosmos DB account "lennidemo". The left sidebar navigation bar is visible, with "Data Explorer" selected. In the main area, the "SQL API" tab is active, and the "Items" section under the "Families" collection is selected. A query titled "Query 1" is running, displaying the results of a SELECT statement:

```
1 SELECT * FROM c
2 WHERE c.address.zipCode = '60601'
```

The results pane shows one item, which is highlighted with a red box. The JSON output for this item is identical to the one shown in the previous screenshot:

```
{
  "familyName": "Smith",
  "address": {
    "addressLine": "123 Main Street",
    "city": "Chicago",
    "state": "IL",
    "zipCode": "60601"
  },
  "parents": [
    "Peter",
    "Alice"
  ],
  "kids": [
    "Adam",
    "Jacqueline",
    "Joshua"
  ]
}
```

The screenshot shows the Microsoft Azure Data Explorer interface for an Azure Cosmos DB account named "lennidemo". The left sidebar navigation bar is visible, with "Data Explorer" selected. The main area displays a query results pane titled "Query 1" with the following SQL query:

```
1 SELECT * FROM c
2 WHERE IS_DEFINED(c.pets)
```

The results pane shows a single item (1 - 1) with the following JSON document:

```
{
  "familyName": "Jones",
  "address": {
    "addressLine": "456 Harbor Boulevard",
    "city": "Chicago",
    "state": "IL",
    "zipCode": "60603"
  },
  "parents": [
    "David",
    "Diana"
  ],
  "kids": [
    "Evan"
  ],
  "pets": [
    "Lint"
  ]
}
```

A red box highlights the WHERE clause of the query and the "pets" array in the JSON result.

This screenshot shows the same Microsoft Azure Data Explorer interface for the "lennidemo" account. The left sidebar navigation bar is visible, with "Data Explorer" selected. The main area displays a query results pane titled "Query 1" with the following SQL query:

```
1 SELECT * FROM c
2 WHERE ARRAY_LENGTH(c.kids) > 2
```

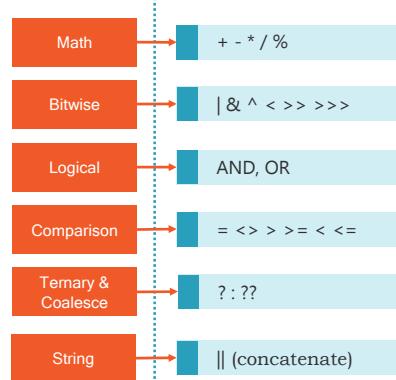
The results pane shows a single item (1 - 1) with the following JSON document:

```
{
  "familyName": "Smith",
  "address": {
    "addressLine": "123 Main Street",
    "city": "Chicago",
    "state": "IL",
    "zipCode": "60601"
  },
  "parents": [
    "Peter",
    "Alice"
  ],
  "kids": [
    "Adam",
    "Jacqueline",
    "Joshua"
  ]
}
```

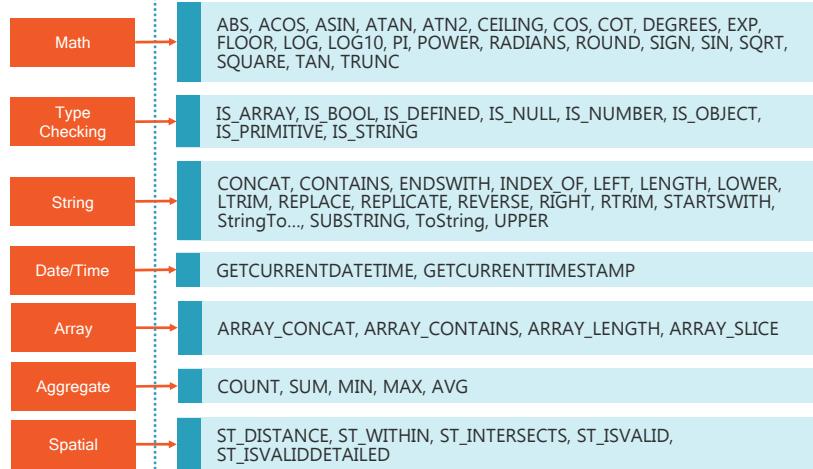
A red box highlights the WHERE clause of the query and the "kids" array in the JSON result.

SQL Operators and Functions

Common operators



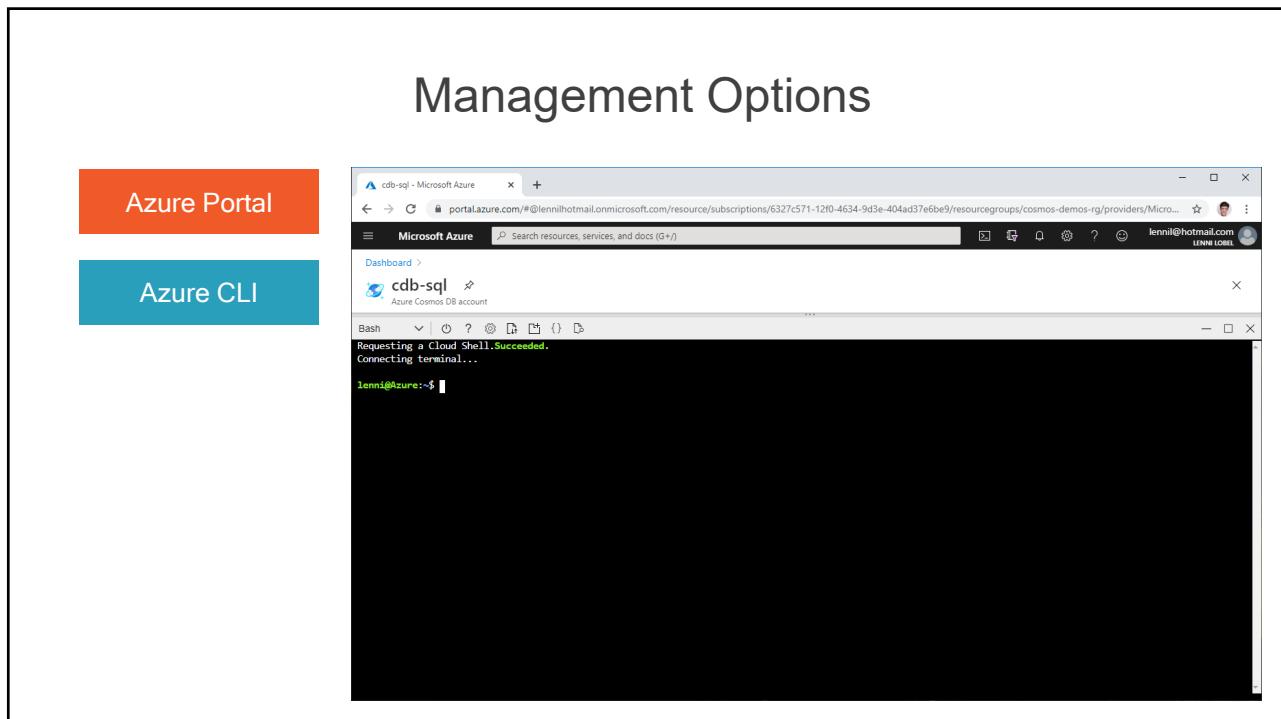
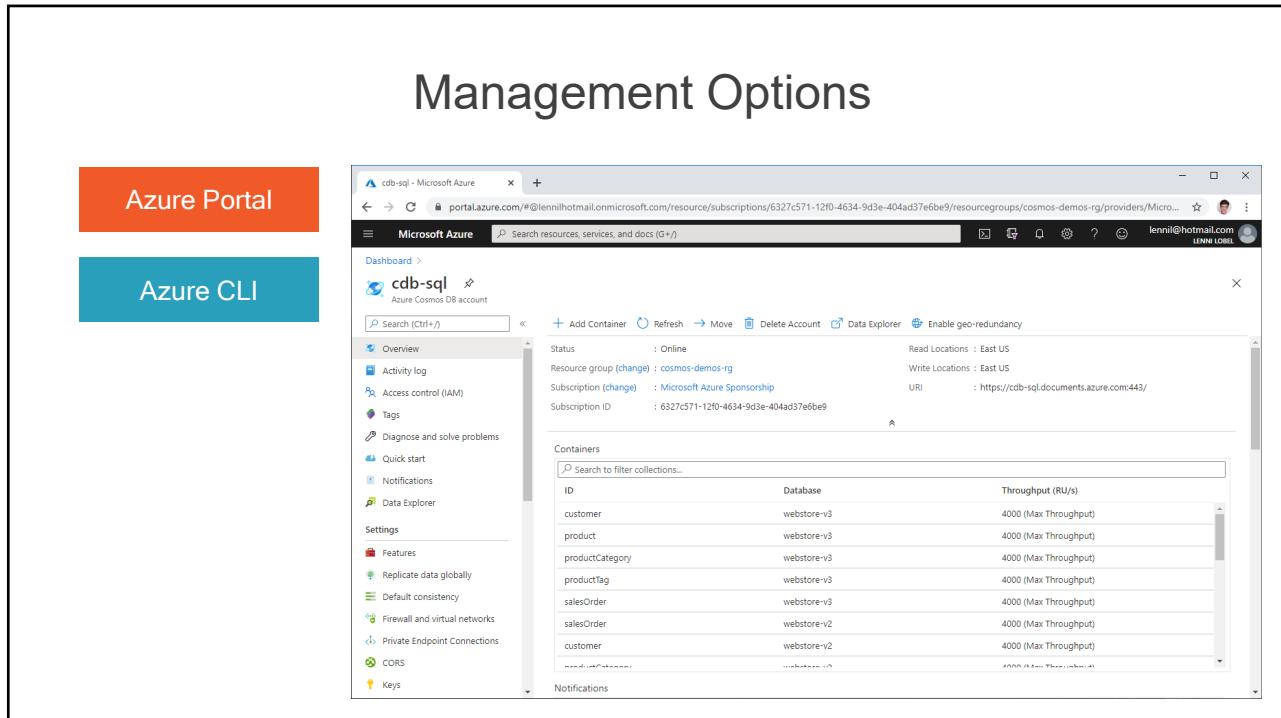
Built-in functions



Introduction to Azure Cosmos DB

Management Options





Management Options

The screenshot displays two management interfaces side-by-side. On the left, there is an orange box labeled "Azure Portal" and a teal box labeled "Azure CLI". To the right of these boxes is a screenshot of a Microsoft Edge browser window titled "cdb-sql - Microsoft Azure". The browser shows the Azure portal for the "cdb-sql" Azure Cosmos DB account. A terminal window is open within the browser, displaying the command-line interface for managing the database. The terminal output shows the creation of a database named "my-database" in the "cosmos-demos-rg" resource group under the "cdb-sql" account.

```

cdb-sql - Microsoft Azure
cdb-sql
Azure Cosmos DB account

Bash
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

lenni@Azure:~$ #!/bin/bash
lenni@Azure:~$ 
lenni@Azure:~$ # Create a database
lenni@Azure:~$ resourceGroupName="cosmos-demos-rg"
lenni@Azure:~$ accountName="cdb-sql"
lenni@Azure:~$ databaseName="my-database"
lenni@Azure:~$ 
lenni@Azure:~$ az cosmosdb sql database create \
>   -a $accountName \
>   -g $resourceGroupName \
>   -n $databaseName

```

Management Options

The screenshot displays three management interfaces side-by-side. On the left, there is an orange box labeled "Azure Portal", a teal box labeled "Azure CLI", and a purple box labeled "PowerShell". To the right of these boxes is a screenshot of a Microsoft Edge browser window titled "cdb-sql - Microsoft Azure". The browser shows the Azure portal for the "cdb-sql" Azure Cosmos DB account. A terminal window is open within the browser, displaying the command-line interface for managing the database. The terminal output shows the creation of a database named "my-database" in the "cosmos-demos-rg" resource group under the "cdb-sql" account. The output also includes a warning about the "cosmosdb sql" command being in preview.

```

cdb-sql - Microsoft Azure
cdb-sql
Azure Cosmos DB account

Bash
lenni@Azure:~$ #!/bin/bash
lenni@Azure:~$ 
lenni@Azure:~$ # Create a database
lenni@Azure:~$ resourceGroupName="cosmos-demos-rg"
lenni@Azure:~$ accountName="cdb-sql"
lenni@Azure:~$ databaseName="my-database"
lenni@Azure:~$ 
lenni@Azure:~$ az cosmosdb sql database create \
>   -a $accountName \
>   -g $resourceGroupName \
>   -n $databaseName
Command group 'cosmosdb sql' is in preview. It may be changed/removed in a future release.
{
  "id": "/subscriptions/6327c571-12f0-4634-9d3e-404ad37e6be9/resourceGroups/cosmos-demos-rg/providers/Microsoft.DocumentDB/databaseAccounts/cdb-sql/sqlDatabases/my-database",
  "location": null,
  "name": "my-database",
  "options": null,
  "resource": {
    "self": " dbs/yqdIAA==/",
    "id": "my-database"
  },
  "resourceGroup": "cosmos-demos-rg",
  "tags": null,
  "type": "Microsoft.DocumentDB/databaseAccounts/sqlDatabases"
}
lenni@Azure:~$ 

```

Management Options

Azure Portal

Azure CLI

PowerShell

```
Administrator: Windows PowerShell
PS C:\windows\system32>
```

Management Options

Azure Portal

Azure CLI

PowerShell

```
Administrator: Windows PowerShell
PS C:\windows\system32> $resourceGroupName = "cosmos-demos-rg"
>> $accountName = "cdb-sql"
>> $databaseName = "my-database"
>>
>> New-AzCosmosDBSqlDatabase `
```

Management Options

Azure Portal

Azure CLI

PowerShell

ARM Templates

```

Administrator: Windows PowerShell
PS C:\windows\system32> $resourceGroupName = "cosmos-demos-rg"
>> $accountName = "cdb-sql"
>> $databaseName = "my-database"
>>
>> New-AzCosmosDBSqlDatabase ` 
>>     -ResourceGroupName $resourceGroupName ` 
>>     -AccountName $accountName ` 
>>     -Name $databaseName
>>
Name      : my-database
Id       : /subscriptions/6327c571-12f0-4634-9d3e-404ad37e6be9/resourceGroups/cosmos-demos-rg/providers/Microsoft.DocumentDB/databaseAccounts/cdb-sql/sqlDatabases/my-database
Location :
Tags    :
Resource : Microsoft.Azure.Commands.CosmosDB.Models.PSSqlDatabaseGetPropertiesResource

PS C:\windows\system32> -

```

Management Options

Azure Portal

Azure CLI

PowerShell

ARM Templates

```
{
  "type": "Microsoft.DocumentDB/databaseAccounts",
  "apiVersion": "2015-04-08",
  "name": "[variables('accountName')]",
  "location": "[resourceGroup().location]",
  "tags": {},
  "kind": "GlobalDocumentDB",
  "properties": {
    "consistencyPolicy": {
      "defaultConsistencyLevel": "Eventual",
      "maxStalenessPrefix": 1,
      "maxIntervalInSeconds": 5
    },
    "locations": [
      {
        "locationName": "West Europe",
        "failoverPriority": 0
      },
      {
        "locationName": "East US 2",
        "failoverPriority": 1
      }
    ],
    "databaseAccountOfferType": "Standard",
    "enableAutomaticFailover": true,
    "enableMultipleWriteLocations": true
  }
}
```

Management Options

The screenshot shows the Azure Portal interface with a sidebar on the left containing colored buttons for different management tools:

- Azure Portal (Orange)
- Azure CLI (Teal)
- PowerShell (Purple)
- ARM Templates (Grey)
- Notebooks (Maroon)

The main area displays the Azure Cosmos DB Data Explorer. A sub-menu for "Data Explorer" is open, showing options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer. The "Data Explorer" option is highlighted.

In the Data Explorer, a "Create Database..." dialog is open with the heading "Create a new database". Below it, there is a code cell labeled "Cell 1" with the placeholder text "Create a new database".

```
1 # Create database
2 database = cosmos_client.create_database('my-database')
3 print('Created database "my-database"')
```

The status bar at the bottom indicates "Last saved 3 minutes".

Management Options

The screenshot shows the Azure Portal interface with a sidebar on the left containing colored buttons for different management tools:

- Azure Portal (Orange)
- Azure CLI (Teal)
- PowerShell (Purple)
- ARM Templates (Grey)
- Notebooks (Maroon)

The main area displays the Azure Cosmos DB Data Explorer. A sub-menu for "Data Explorer" is open, showing options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Notifications, and Data Explorer. The "Data Explorer" option is highlighted.

In the Data Explorer, a "Create Database..." dialog is open with the heading "Create a new database". Below it, there is a code cell labeled "Cell 1" with the placeholder text "Create a new database".

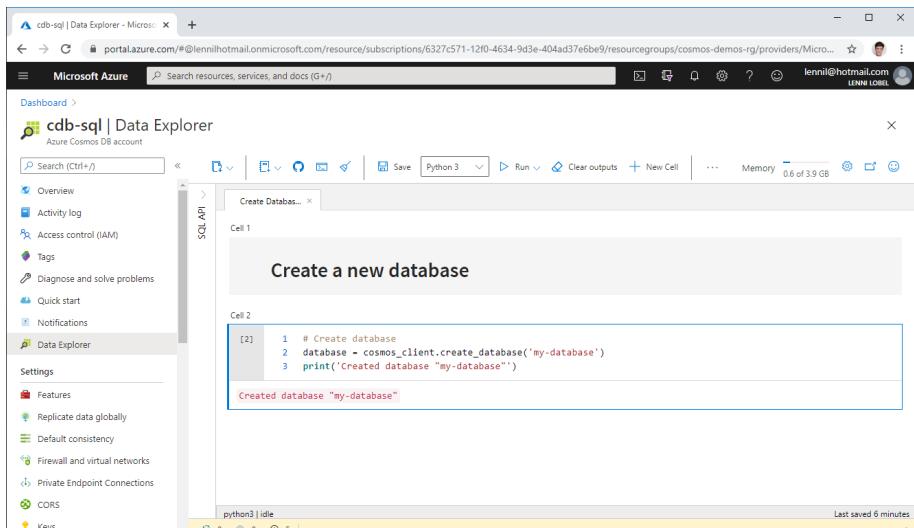
```
1 # Create database
2 database = cosmos_client.create_database('my-database')
3 print('Created database "my-database"')
```

In the code cell labeled "Cell 2", the same Python code is shown:

```
1 # Create database
2 database = cosmos_client.create_database('my-database')
3 print('Created database "my-database"')
```

The status bar at the bottom indicates "Last saved 3 minutes".

Management Options

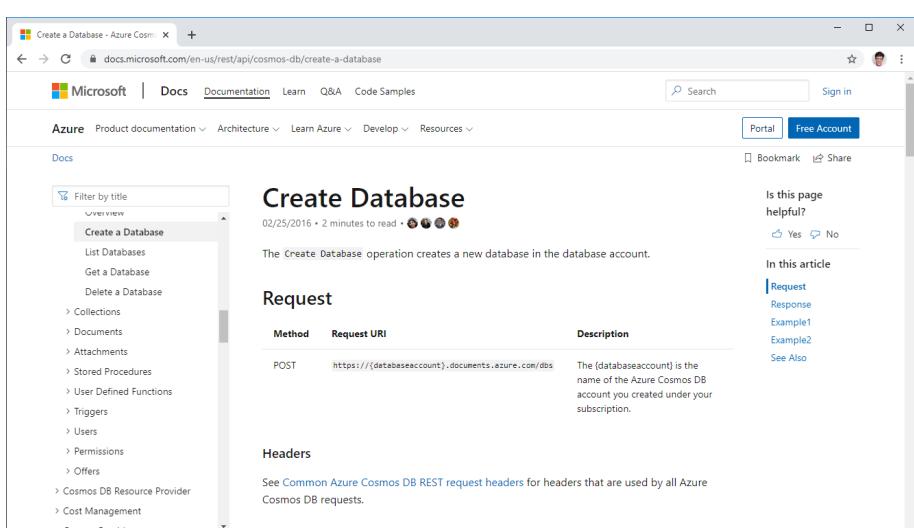


The screenshot shows the Azure Portal's Data Explorer interface. On the left, a sidebar lists various management options: Azure Portal, Azure CLI, PowerShell, ARM Templates, Notebooks, and REST API. The main area displays a Python notebook cell titled "Create a new database". The cell contains the following code:

```
[2]: 1 # Create database
2 database = cosmos_client.create_database('my-database')
3 print('Created database "my-database"')
```

The output of the cell shows the message "Created database "my-database"".

Management Options



The screenshot shows a Microsoft Docs page titled "Create a Database - Azure Cosmos DB". The left sidebar lists management options: Azure Portal, Azure CLI, PowerShell, ARM Templates, Notebooks, and REST API. The main content area is titled "Create Database" and provides documentation for creating a new database using the REST API. It includes sections for "Request", "Description", "Headers", and "Body".

Create Database

The `Create Database` operation creates a new database in the database account.

Request

Method	Request URI	Description
POST	<code>https://(databaseaccount).documents.azure.com/dbs</code>	The <code>(databaseaccount)</code> is the name of the Azure Cosmos DB account you created under your subscription.

Headers

See [Common Azure Cosmos DB REST request headers](#) for headers that are used by all Azure Cosmos DB requests.

Body

Introduction to Azure Cosmos DB

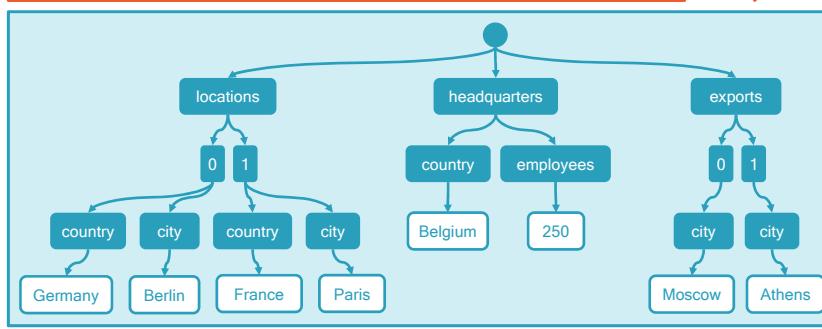
Automatic Indexing



Automatic Indexing

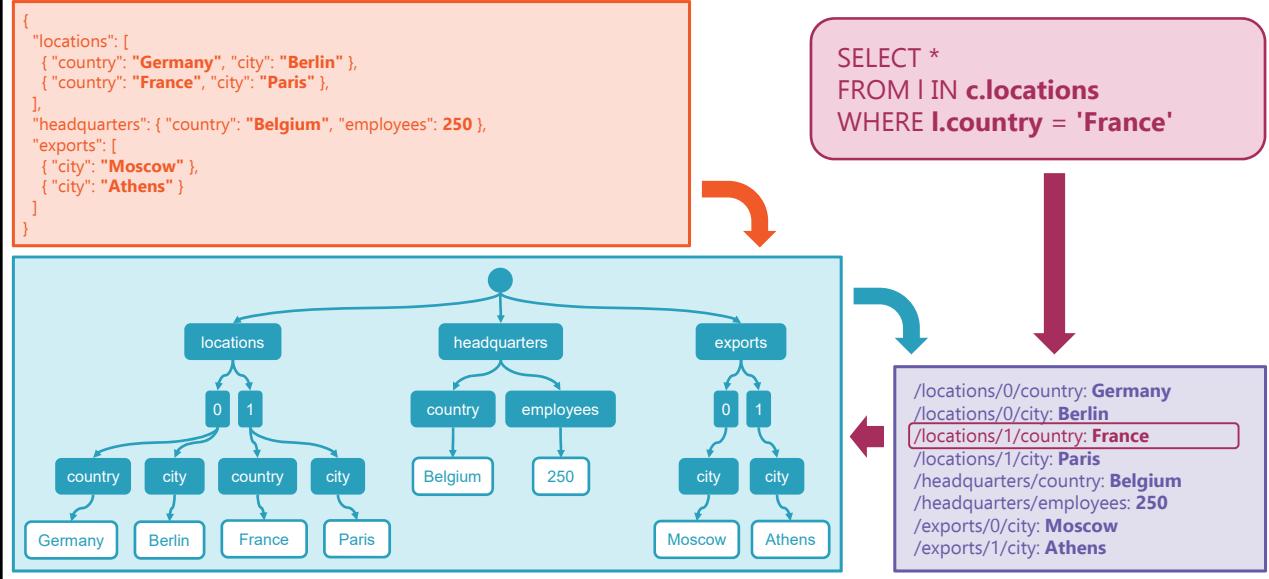
```
{
  "locations": [
    { "country": "Germany", "city": "Berlin" },
    { "country": "France", "city": "Paris" }
  ],
  "headquarters": { "country": "Belgium", "employees": 250 },
  "exports": [
    { "city": "Moscow" },
    { "city": "Athens" }
  ]
}
```

SELECT *
FROM l IN c.locations
WHERE l.country = 'France'

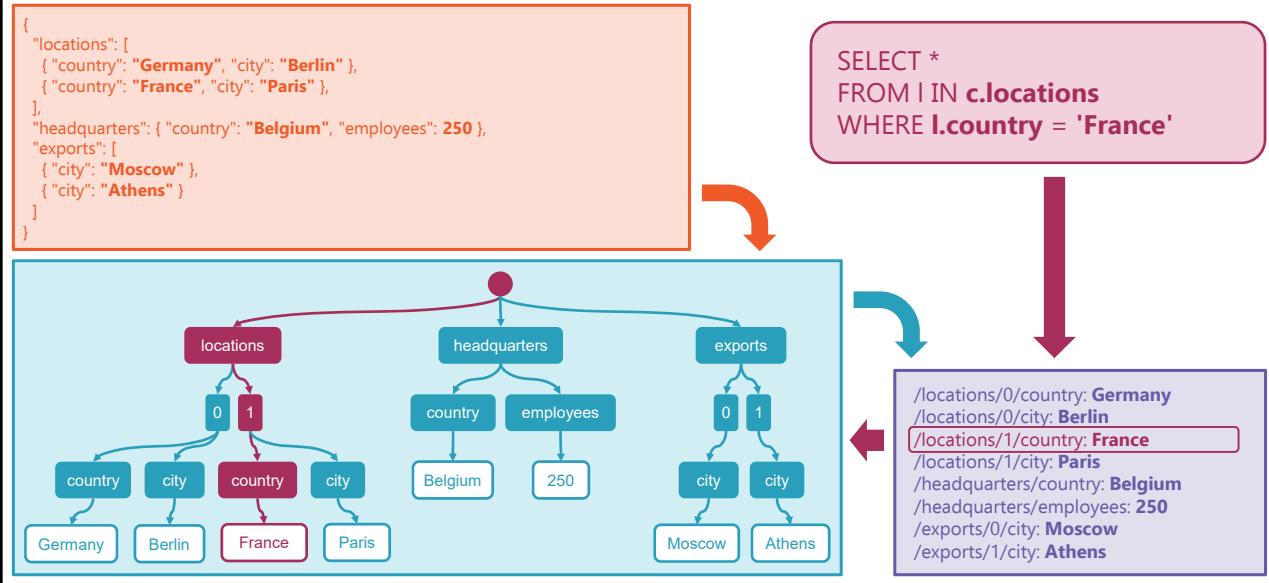


/locations/0/country: **Germany**
/locations/0/city: **Berlin**
/locations/1/country: **France**
/locations/1/city: **Paris**
/headquarters/country: **Belgium**
/headquarters/employees: **250**
/exports/0/city: **Moscow**
/exports/1/city: **Athens**

Automatic Indexing



Automatic Indexing

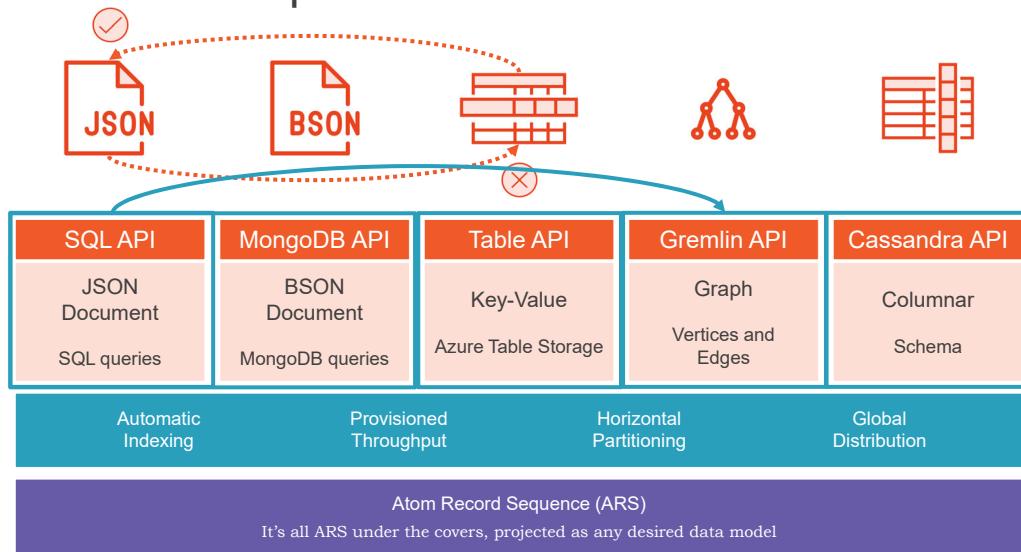


Introduction to Azure Cosmos DB

Multiple APIs and Data Models



Multiple APIs and Data Models



Multiple APIs and Data Models



SQL API	MongoDB API	Table API	Gremlin API	Cassandra API
JSON Document	BSON Document	Key-Value	Graph Vertices and Edges	Columnar Schema
SQL queries	MongoDB queries	Azure Table Storage		
Automatic Indexing	Provisioned Throughput	Horizontal Partitioning	Global Distribution	
Atom Record Sequence (ARS) It's all ARS under the covers, projected as any desired data model				



Introduction to Azure Cosmos DB

Data Migration Options



Data Migration Options

Data Migration Tool

Open source desktop application, suitable for small-to-medium datasets

- Raw JSON files
- MongoDB
- SQL Server
- Raw CSV files
- Azure Table Storage
- Amazon DynamoDB
- HBase
- Cosmos DB

<https://docs.microsoft.com/azure/cosmos-db/import-data>

<https://github.com/azure/azure-documentdb-datamigrationtool>



Data Migration Options

Data Migration Tool

Open source desktop application, suitable for small-to-medium datasets

Azure Data Factory

ETL service for data integration, suitable for medium-to-large datasets

<https://docs.microsoft.com/azure/data-factory>



Data Migration Options

Data Migration Tool

Open source desktop application, suitable for small-to-medium datasets

Azure Data Factory

ETL service for data integration, suitable for medium-to-large datasets

Spark connector

Big data analytics using Apache Spark

<https://docs.microsoft.com/azure/cosmos-db/spark-connector>



Data Migration Options

Data Migration Tool

Open source desktop application, suitable for small-to-medium datasets

Azure Data Factory

ETL service for data integration, suitable for medium-to-large datasets

Spark connector

Big data analytics using Apache Spark

Custom tool

Support extremely large (10 TB+) datasets

<https://docs.microsoft.com/en-us/azure/cosmos-db/migrate-cosmosdb-data>



Data Migration Options

Data Migration Tool

Open source desktop application, suitable for small-to-medium datasets

Azure Data Factory

ETL service for data integration, suitable for medium-to-large datasets

Spark connector

Big data analytics using Apache Spark

Custom tool

Support extremely large (10 TB+) datasets

Online migrations

Change feed
Striim

<https://docs.microsoft.com/en-us/azure/cosmos-db/change-feed>

<https://docs.microsoft.com/en-us/azure/cosmos-db/cosmosdb-sql-api-migrate-data-striim>



Data Migration Options

Data Migration Tool

Open source desktop application, suitable for small-to-medium datasets

Azure Data Factory

ETL service for data integration, suitable for medium-to-large datasets

Spark connector

Big data analytics using Apache Spark

Custom tool

Support extremely large (10 TB+) datasets

Online migrations

Change feed
Striim

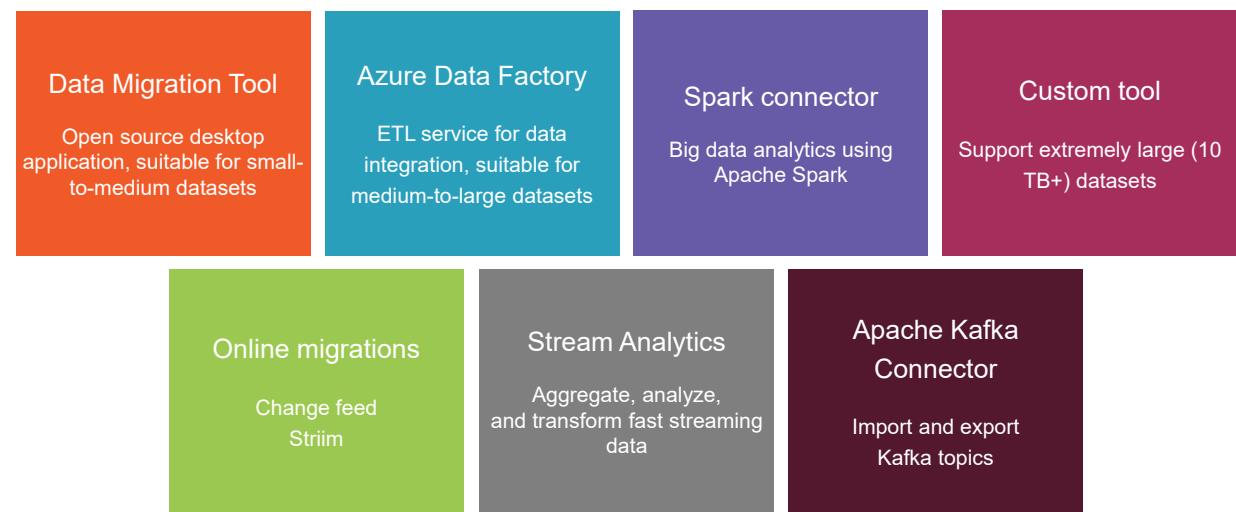
Stream Analytics

Aggregate, analyze, and transform fast streaming data

<https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-documentdb-output>



Data Migration Options



<https://docs.microsoft.com/en-us/azure/cosmos-db/sql/kafka-connector-sink>



Introduction to Azure Cosmos DB

Throughput and Cost



Measuring Performance

Latency

How fast is the response
for a
given request?

Throughput

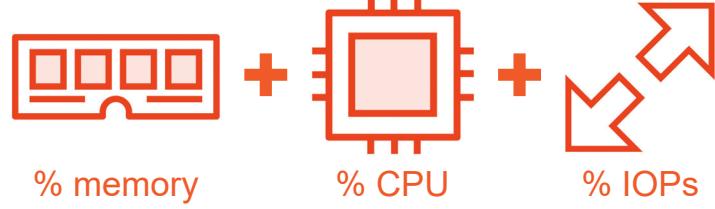
How many requests can
be served within a
specific period of time?



Introducing Request Units

Throughput Currency

Blended measure of
computational cost
(memory, CPU, IOPs)



Introducing Request Units

Throughput Currency

Blended measure of computational cost (memory, CPU, IOPs)

All Requests are Not Equal

Every Cosmos DB response header shows the RU charge for the request

Request Units are Deterministic

The same request will always require the same number of request units



Monitoring Request Unit Consumption

Metric	Value
Request Charge	2.83 RUs
Showing Results	1 - 1
Retrieved document count	1
Retrieved document size	436 bytes

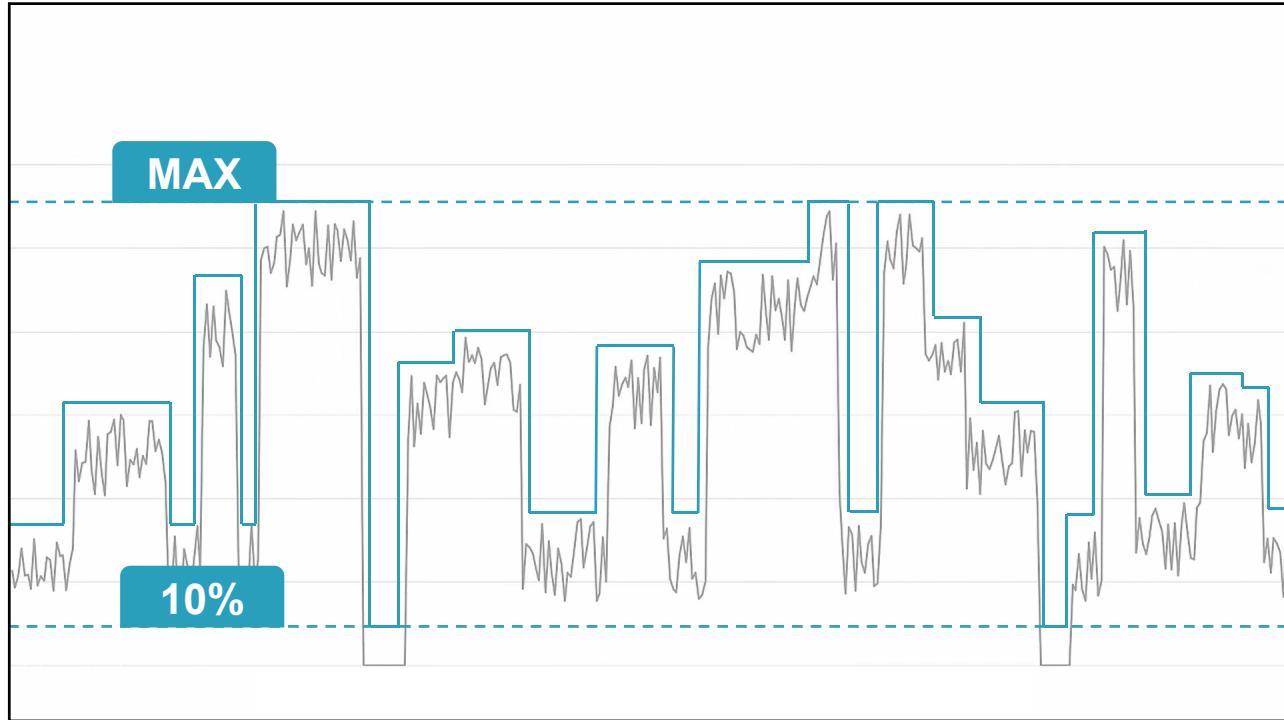
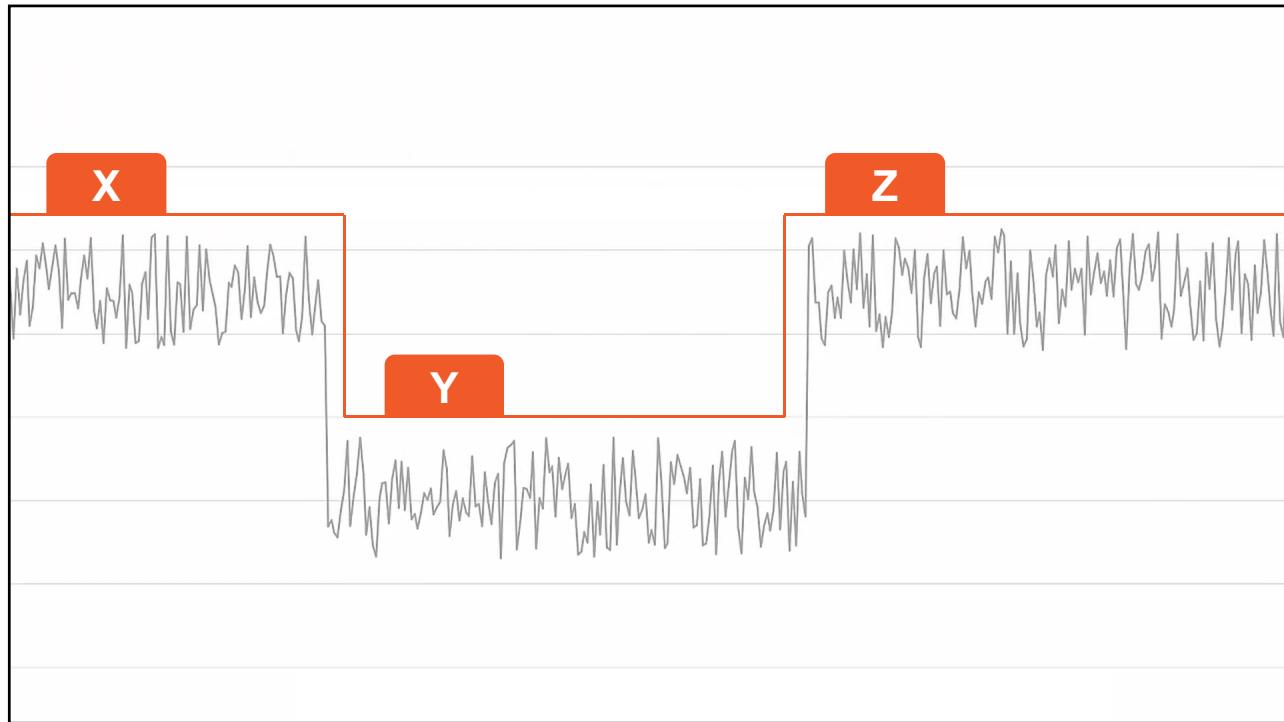
Metric	Value
Request Charge	3.13 RUs
Showing Results	1 - 1
Retrieved document count	1
Retrieved document size	436 bytes

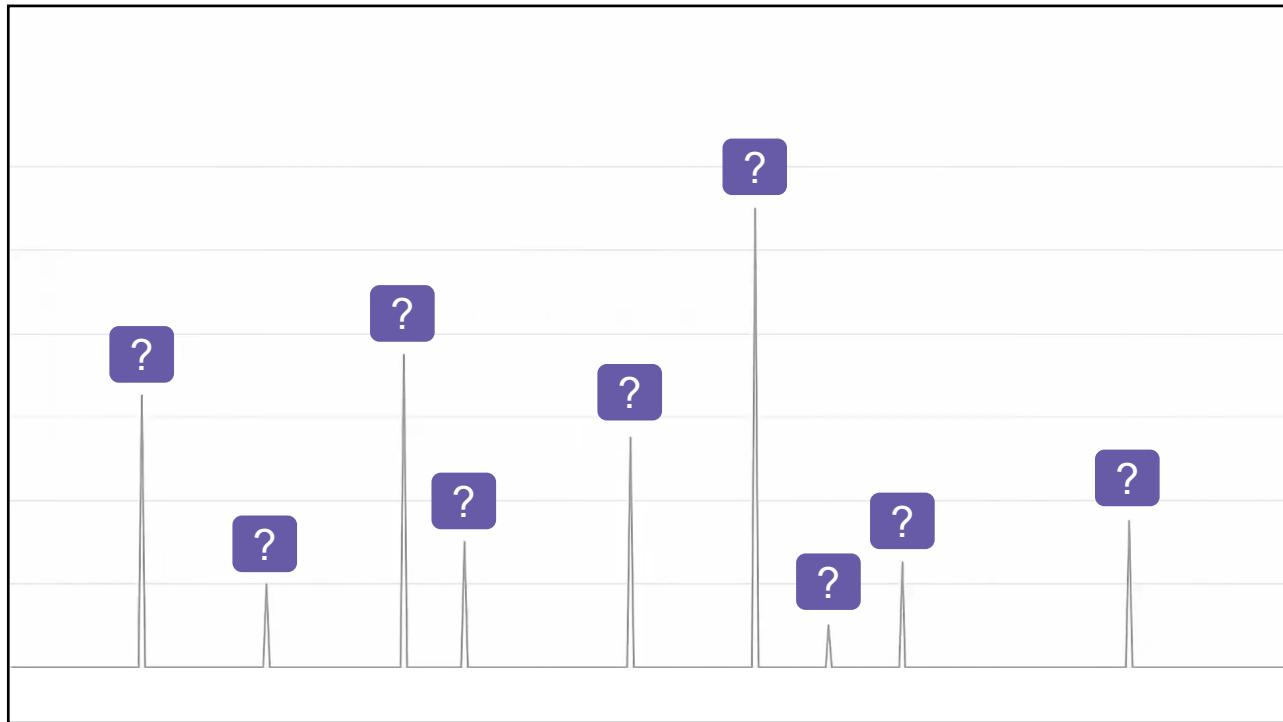
Visual Studio Live! Austin 2022

The screenshot shows the Azure Metrics (Classic) dashboard for the 'cosmos-pluralsight-course' account. The left sidebar includes links for Document Explorer, Query Explorer, Script Explorer, Monitoring (with Insights, Alerts, Metrics, Logs, Diagnostic settings), Metrics (Classic), Workbooks, Automation (Tasks (preview), Export template), and Support + troubleshooting (New Support Request). The main area has tabs for Overview, Throughput, Storage, Availability, Latency, Consistency, and System. Under Throughput, there are two charts: 'Number of requests (aggregated over 1 minute interval)' showing a sharp peak at 8:15 AM with values for Total, Http 2xx, Http 400, Http 401, Http 403, Http 429, and Http 5xx; and 'Number of requests exceeded capacity (aggregated over 1 minute interval)' showing 'Http 429' with a note 'No data to display'.

The screenshot shows the Visual Studio code editor with the file 'Program.cs' open. The code creates a new CosmosClient, gets a container for 'Families', and defines a dynamic document representing a family. The document includes fields for id, familyName, address (with addressLine, city, state, zipCode), parents (an array containing 'Peter' and 'Alice'), and kids (an array containing 'Adam', 'Jacqueline', and 'Joshua'). It then uses the container's CreateItemAsync method to insert the document and prints the consumed RUs. A tooltip for 'consumedRUs' shows a value of 9.9.

```
25 var client = new CosmosClient(endpoint, masterKey);
26 var container = client.GetContainer("Families", "Families");
27
28 dynamic document = new
29 {
30     id = Guid.NewGuid(),
31     familyName = "Smith",
32     address = new
33     {
34         addressLine = "123 Main Street",
35         city = "Chicago",
36         state = "IL",
37         zipCode = "60601"
38     },
39     parents = new string[]
40     {
41         "Peter",
42         "Alice"
43     },
44     kids = new string[]
45     {
46         "Adam",
47         "Jacqueline",
48         "Joshua"
49     }
50 };
51
52 var result = await container.CreateItemAsync(document, new PartitionKey(document.address.zipCode));
53 var consumedRUs = result.RequestCharge;
54
55 Console.WriteLine($"Cost to create document: {consumedRUs} RUs");
56
57 }
```





Throughput Offers

Provisioned throughput (manual)

Reserve RU/sec

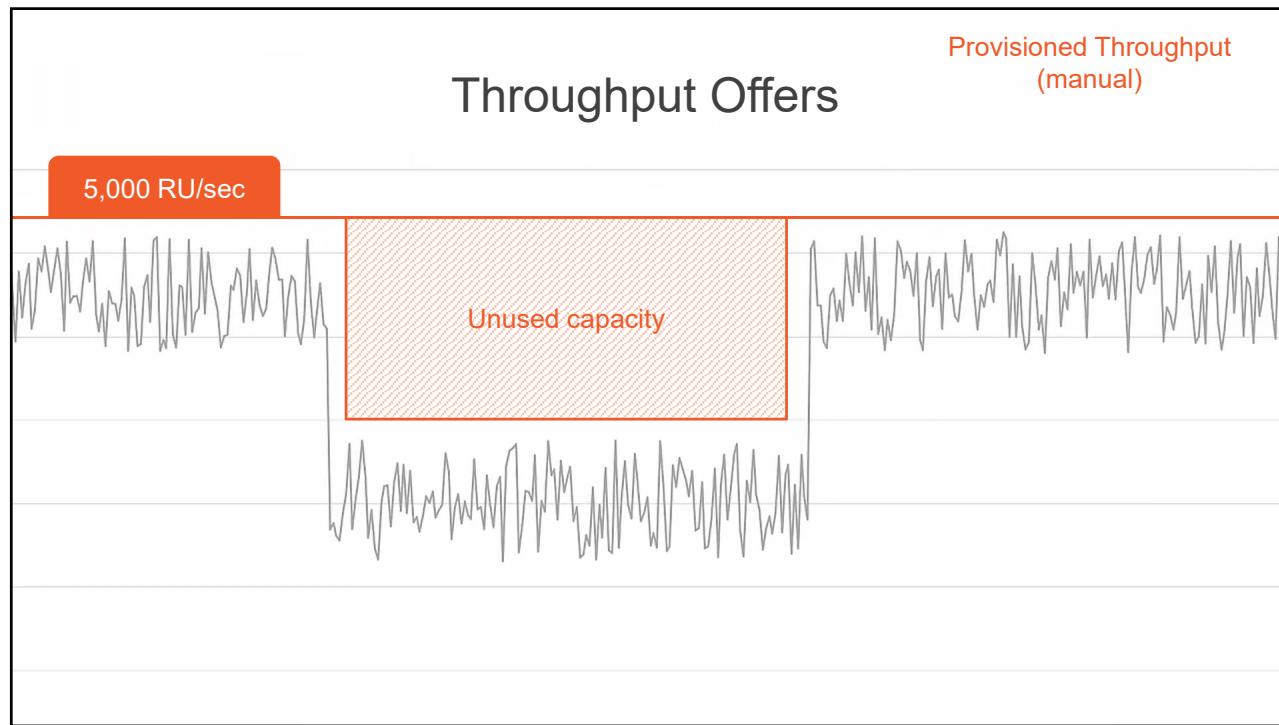
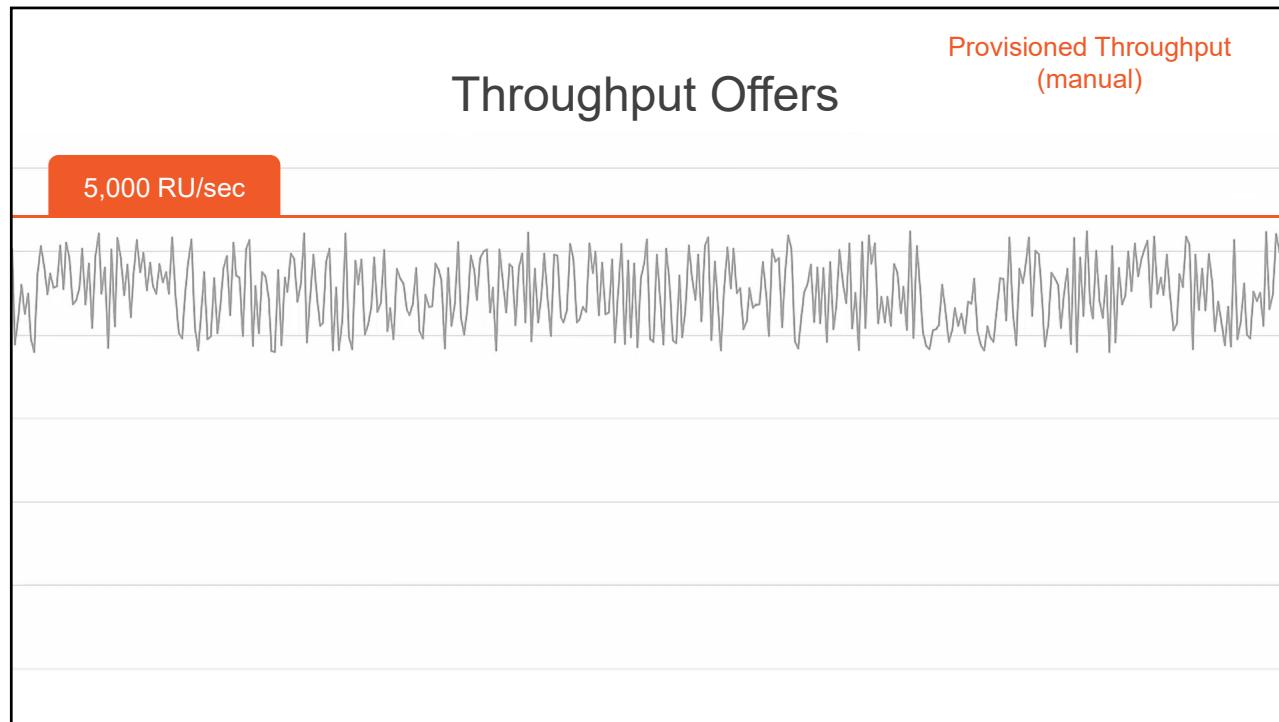
Guaranteed always available

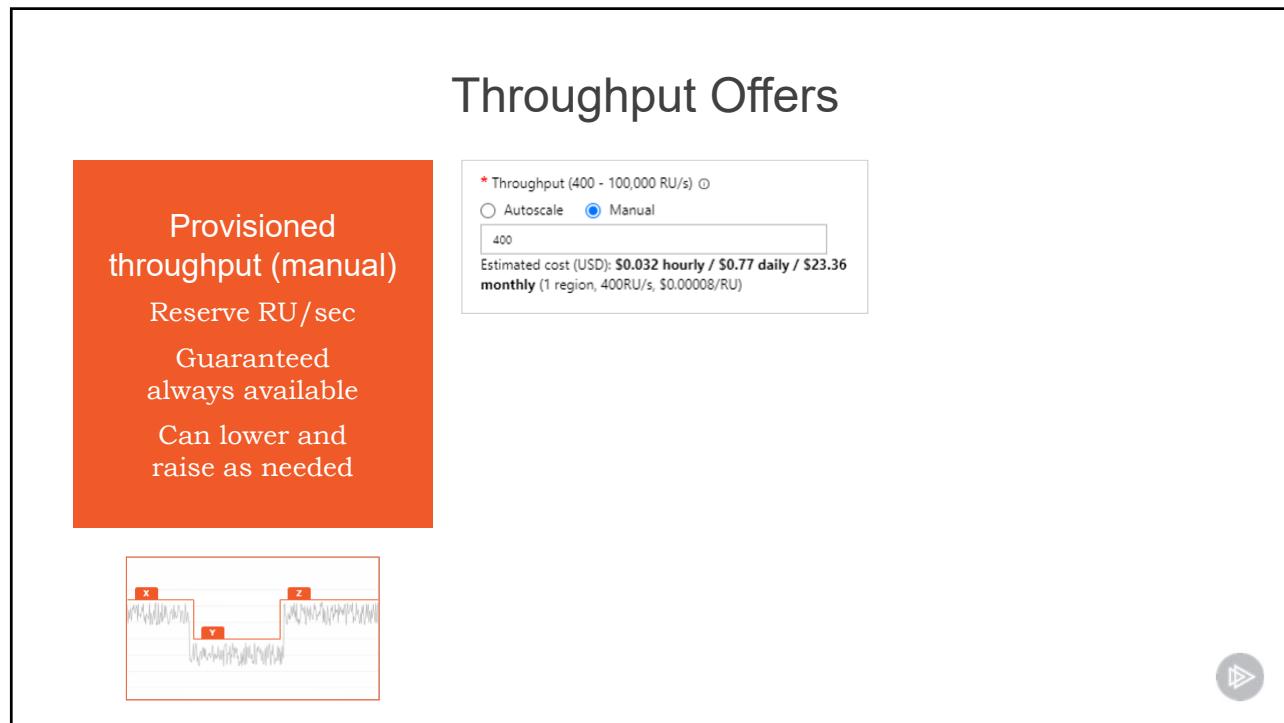
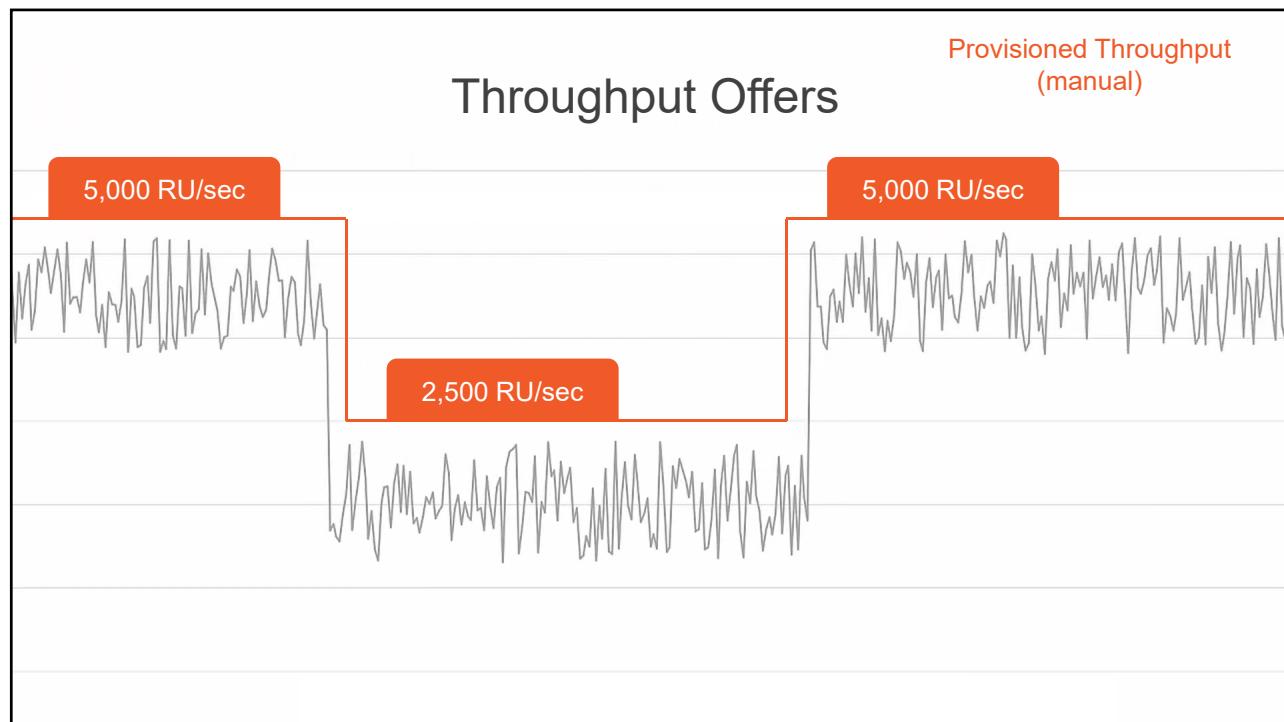
Can lower and raise as needed

* Throughput (400 - 100,000 RU/s) Autoscale Manual
400

Estimated cost (USD): \$0.032 hourly / \$0.77 daily / \$23.36 monthly (1 region, 400RU/s, \$0.00008/RU)







Throughput Offers

Provisioned throughput (manual)

Reserve RU/sec

Guaranteed always available

Can lower and raise as needed

Provisioned throughput (autoscale)

Reserve max RU/sec

Scales up and down automatically

From max to 10% below

* Throughput (autoscale)

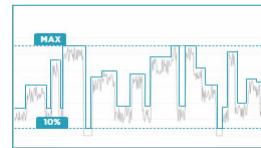
Autoscale Manual

Provision maximum RU/s required by this resource. Estimate your required RU/s with [capacity calculator](#).

Max RU/s

4000

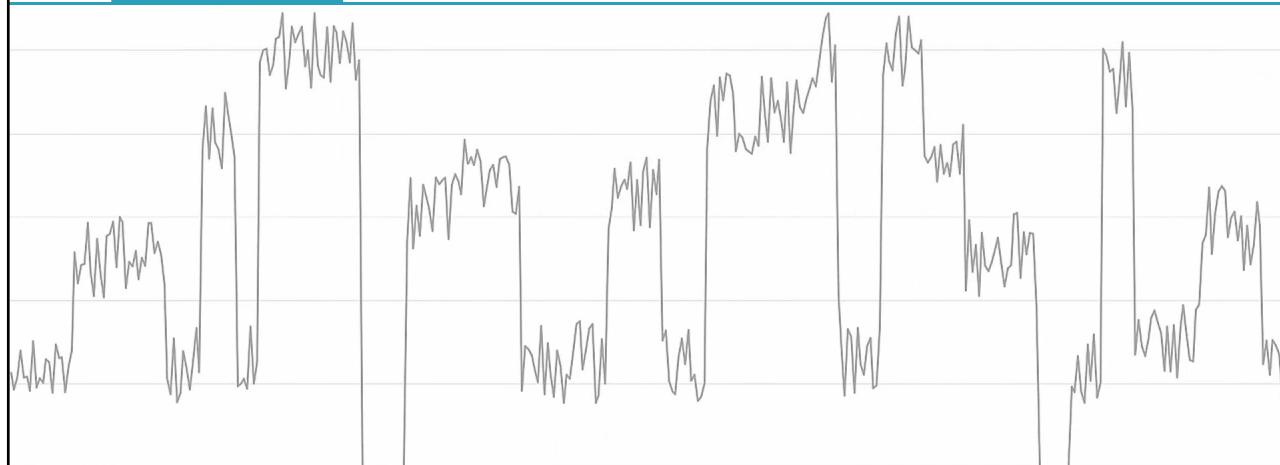
Your container throughput will automatically scale from **400 RU/s** (**10% of max RU/s**) - **4000 RU/s** based on usage.



Provisioned Throughput
(autoscale)

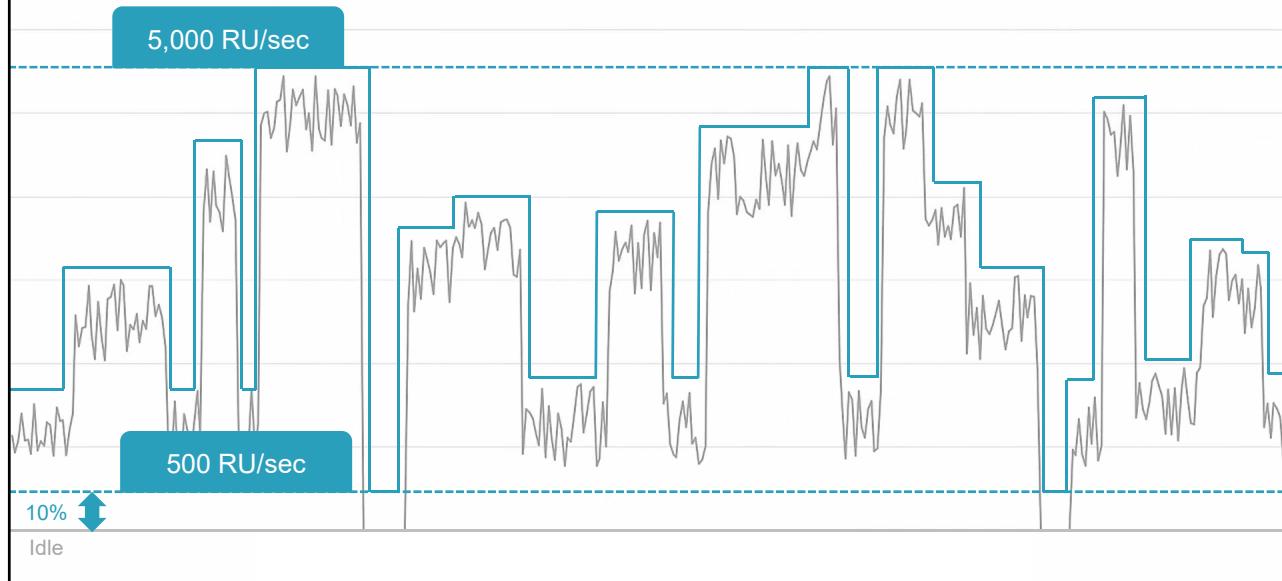
Throughput Offers

? RU/sec



Throughput Offers

Provisioned Throughput
(autoscale)



Throughput Offers

Provisioned throughput (manual)

Reserve RU/sec

Guaranteed always available

Can lower and raise as needed

Provisioned throughput (autoscale)

Reserve max RU/sec

Scales up and down automatically

From max to 10% below

* Throughput (autoscale)

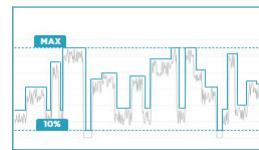
Autoscale Manual

Provision maximum RU/s required by this resource. Estimate your required RU/s with [capacity calculator](#).

Max RU/s

4000

Your container throughput will automatically scale from **400 RU/s** (10% of max RU/s) - **4000 RU/s** based on usage.



Throughput Offers

Provisioned throughput (manual)

Reserve RU/sec

Guaranteed always available

Can lower and raise as needed

Provisioned throughput (autoscale)

Reserve max RU/sec

Scales up and down automatically

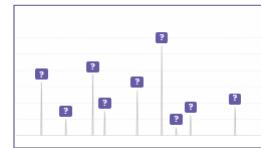
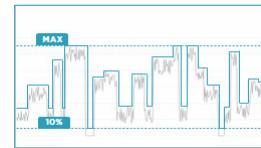
From max to 10% below

Serverless

Consumption model

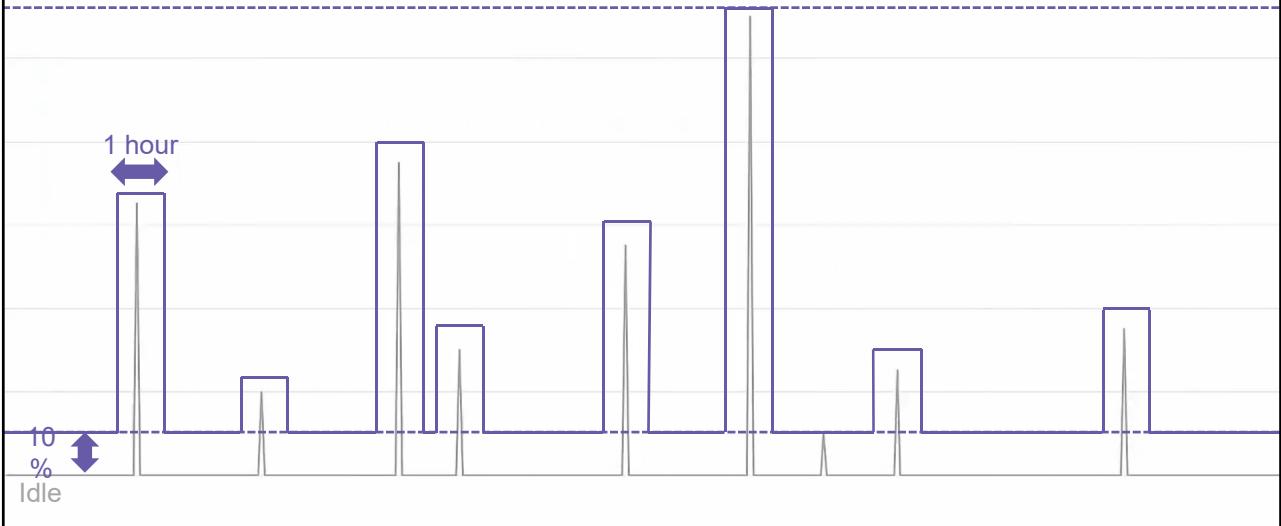
No provisioned throughput

Pay only for the RUs that you use



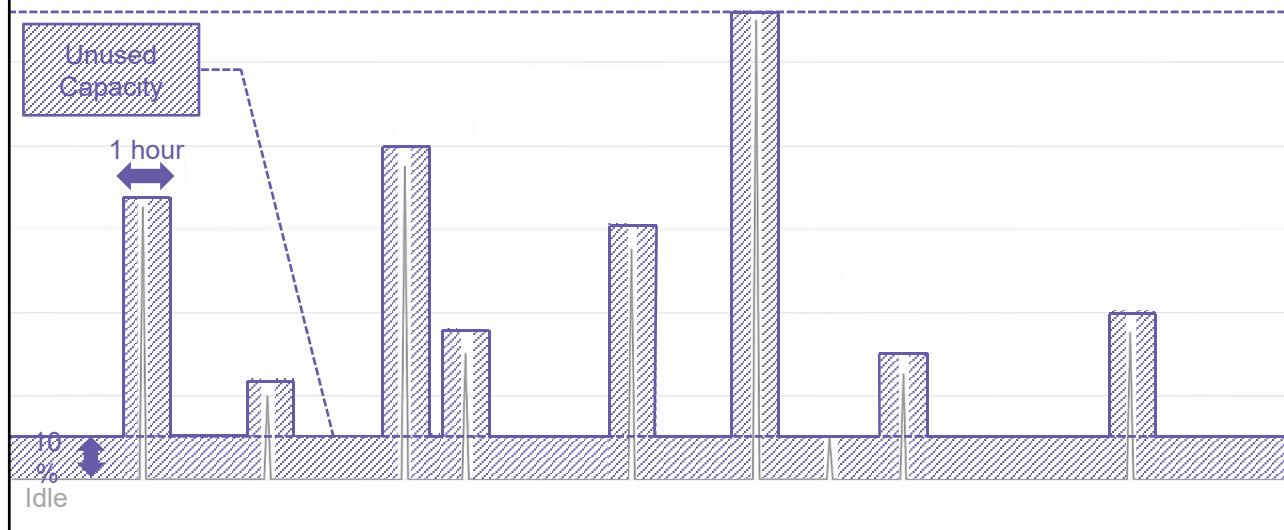
Serverless

Throughput Offers



Serverless

Throughput Offers



Serverless

Throughput Offers



Provisioning Container Throughput

Reserve request units per second (RU/s)

How many request *units* (not *requests*) per second are available to your application

Exceeding reserved throughput limits

Requests are “throttled” (HTTP 429)

```
HTTP/1.1 429 Too Many Requests
Content-Type: application/json
Server: Microsoft-HTTPAPI/2.0
Access-Control-Allow-Origin: https://cosmos.azure.com
Access-Control-Allow-Credentials: true
x-ms-retry-after-ms: 2853
lsn: 38869
x-ms-schemaversion: 1.8
x-ms-substatus: 3200
Vary: Accept-Encoding
```



Exceeding Provisioned Throughput

The screenshot shows a code editor with C# code attempting to create a document in a Cosmos DB container. A break point is set at line 103. The code uses the `CreateItemAsync` method.

```

100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
    try
    {
        await container.CreateItemAsync(docDef, new PartitionKey(docDef.pk));
    }
    catch (CosmosException ex) when (ex.StatusCode == HttpStatusCode.TooManyRequests) // 429
    {
        Console.WriteLine("Can't create document; request was throttled");
    }

```

A **QuickWatch** window is open, showing the exception object with its properties:

Name	Value	Type
ActivityId	"4d1702db-2c82-4730-9e1e-...	string
Data	(System.Collections.ListDictionaryValue) { }	System.Collections.ListDictionaryValue
HRESULT	-214623008	int
HelpLink	null	string
InnerException	null	System.Exception
Message	"Response status code does not indicate success: 429 Substatus: 3200 Reason: ."	string
RequestCount	0.38	double
Response	null	string
Source	"Microsoft.Azure.Cosmos.C...	string
StackTrace	" at Microsoft.Azure.Cosmos...	string
StatusCodes	TooManyRequests	System.Exception
SubStatus	3200	int

An **Exception Thrown** dialog is also visible, displaying the exception details:

Microsoft.Azure.Cosmos.CosmosException: 'Response status code does not indicate success: 429 Substatus: 3200 Reason: .'

[View Details](#) | [Copy Details](#) | [Start Live Share session...](#)

Exception Settings

Break when this exception type is thrown
Except when thrown from:
 ThroughputTest.dll

[Open Exception Settings](#) | [Edit Conditions](#)



Exceeding Provisioned Throughput

```

100
101
102
103     try
104     {
105         await container.CreateItemAsync(docDef, new PartitionKey(docDef.pk));
106     }
107     catch (CosmosException ex) when (ex.StatusCode == HttpStatusCode.TooManyRequests) // 429
108     {
109         Console.WriteLine("Can't create document; request was throttled");
110     }
111
112
113
114
115
116
117
118
119
120
121
122
123
124

```



Progress Telerik Fiddler Web Debugger

File Edit Rules Tools View Help

WinConfig Replay Go Stream Decode | Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff

#	Result	Protocol	Host	URL
1889	200	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/pranges
1890	429	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1891	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=cab11a89-ce67-4c34-a6b2
1892	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=1b9a3857-a457-4907-aba
1893	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=cab11a89-ce67-4c34-a6b2
1894	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=000118e7-bcaf-4c50-0000
1895	200	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1896	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=1b9a3857-a457-4907-aba
1897	200	HTTP	Tunnel to outlook.office365.com:443	
1898	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=000118e7-bcaf-4c50-0000
1899	200	HTTP	Tunnel to dc.services.visualstudio.com:443	
1900	200	HTTPS	dc.services.visualst...	/v2/track
1901	200	HTTPS	dc.services.visualst...	/v2/track
1902	400	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1903	200	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/pranges
1904	429	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1905	200	HTTPS	management.azure...	/subscriptions/3b427ab5-6c17-4169-9ef9-fd9943a0e5
1906	200	HTTPS	management.azure...	/subscriptions/3b427ab5-6c17-4169-9ef9-fd9943a0e5
1907	200	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1908	429	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1909	200	HTTPS	cosmos-demos.doc...	/ dbs/throughput-test/colls/test-container/docs
1910	200	HTTP	Tunnel to outlook.office365.com:443	
1911	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=000118e7-bcaf-4c50-0000
1912	200	HTTPS	outlook.office365.com	/api/emsmdb/?MailboxId=1b9a3857-a457-4907-aba
1913	204	HTTPS	portal.azure.com	/api/extensiolemetry
1914	200	HTTPS	dc.services.visualst...	/v2/track
1915	200	HTTPS	southcentralus.noti...	/users/80rgid:151a730c-aa4c-4a4f-919d-34f24f742f
1916	200	HTTPS	vortex.data.micros...	/collect/v1

Fiddler Orchestra Beta

Statistics Inspectors AutoResponder Composer

Headers TextView SyntaxView WebForms HexView Auth Cookies Raw

JSON XML

POST https://cosmos-demos.documents.azure.com/dbs/throughput-test/colls/1

Host: cosmos-demos.documents.azure.com

Connection: keep-alive

Content-Length: 55

Origin: https://cosmos.azure.com

X-ms-documentdb-query-enable-scan: true

Authorization: type=3pmaster%26ver=301.0%26sig=%3DA1R8DVsdEo25h6TK4%2F5Cd:

X-ms-documentdb-populatequerymetrics: true

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36

X-ms-documentdb-query-parallelizecrosspartitionquery: true

X-ms-documentdb-query-enablecrosspartition: true

<

>

Find... (press Ctrl+Enter to highlight all) View in Notepad

Transformer Headers TextView SyntaxView ImageView HexView WebView Auth

Caching Cookies Raw JSON XML

HTTP/1.1 429 Too Many Requests

Content-Type: application/json

Server: Microsoft-HTTPAPI/2.0

Access-Control-Allow-Origin: https://cosmos.azure.com

Access-Control-Allow-Credentials: true

X-ms-retry-after-ms: 2853

lsn: 38069

X-ms-schemaVersion: 1.8

X-ms-substatus: 3200

X-ms-xp-rce: 1

X-ms-global-committed-lsn: 38068

X-ms-number-of-read-regions: 0

X-ms-transport-request-id: 2

X-ms-cosmos-lsn: 38069

X-ms-request-charge: 0.38

X-ms-serviceversion: version=2.4.0.0

X-ms-activity-id: a6d0a9b-a181-4c83-82fa-871d0d5bd208

Strict-Transport-Security: max-age=31536000

Find... (press Ctrl+Enter to highlight all) View in Notepad

All Processes 1 / 5,578 https://cosmos-demos.documents.azure.com/dbs/throughput-test/colls/test-container/docs

The screenshot shows the Microsoft Azure Metrics dashboard for the 'cdb-sql' Azure Cosmos DB account. The 'Throughput' tab is selected. Two charts are displayed:

- Number of requests (aggregated over 1 minute interval)**: Shows a total number of requests. The chart shows a significant spike from approximately 6K to 7K between 1:30 PM and 1:45 PM. The legend includes: Total (blue), Http 2xx (light blue), Http 400 (dark blue), Http 401 (purple), Http 403 (teal), Http 429 (yellow), and Http 5xx (grey).
- Number of requests exceeded capacity (aggregated over 1 minute interval)**: Shows requests exceeding capacity. A single yellow line represents 'Http 429' requests, which spikes to about 400 at 1:45 PM.

Filtering options include Database(s), Container(s), and Region(s). The time range is set from 07/09/2020 1:21:00 PM to 07/09/2020 1:50:00 PM.

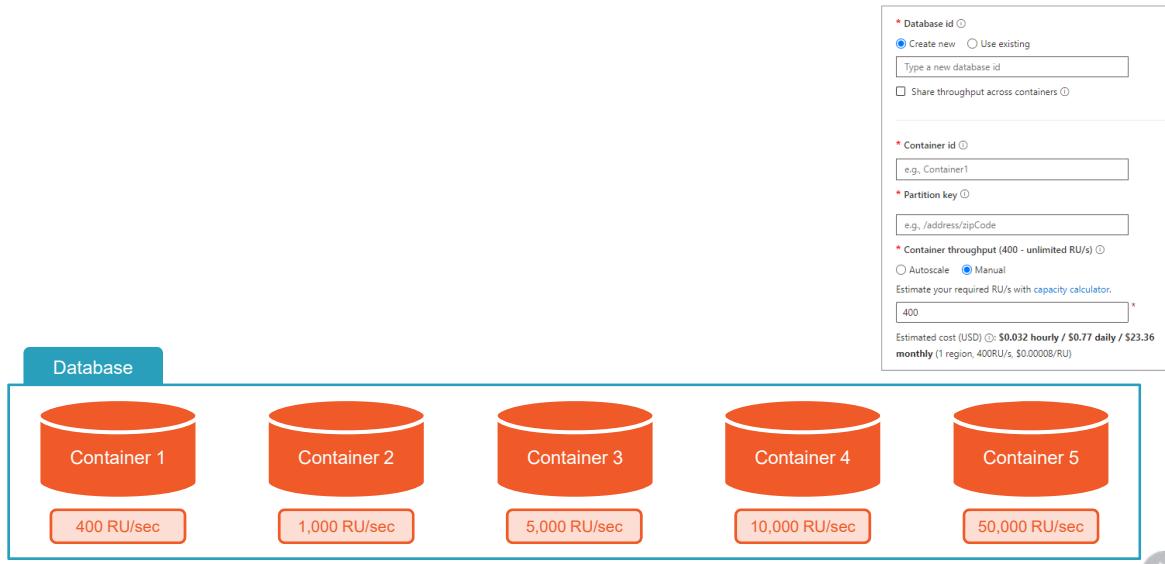
Provisioning Database Throughput

This dialog box is used for provisioning database throughput. It includes the following fields:

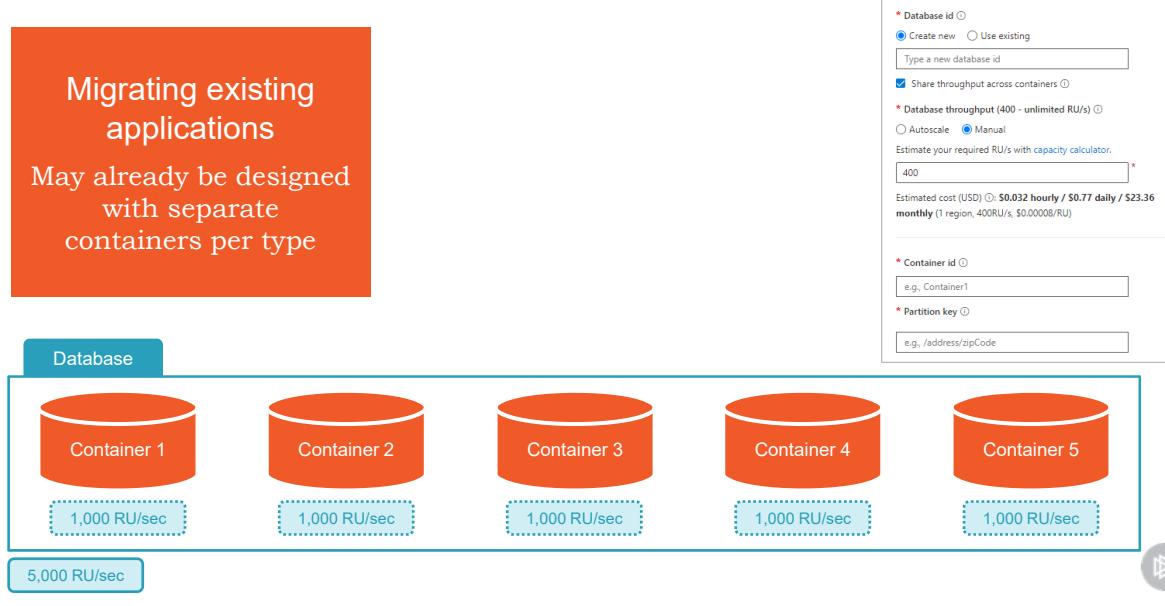
- Database id**:
 - Create new
 - Use existing
- Type a new database id:
- Share throughput across containers
- Database throughput (400 - unlimited RU/s)**:
 - Autoscale
 - Manual
- Estimate your required RU/s with [capacity calculator](#).
- 400
- Estimated cost (USD): \$0.032 hourly / \$0.77 daily / \$23.36 monthly (1 region, 400RU/s, \$0.00008/RU)
- Container id**: e.g., Container1
- Partition key**: e.g., /address/zipCode



Provisioning Database Throughput



Provisioning Database Throughput



Provisioning Database Throughput

Migrating existing applications

May already be designed with separate containers per type

Differentiate solely on partition key

Containers share throughput needs, but different partitioning requirements

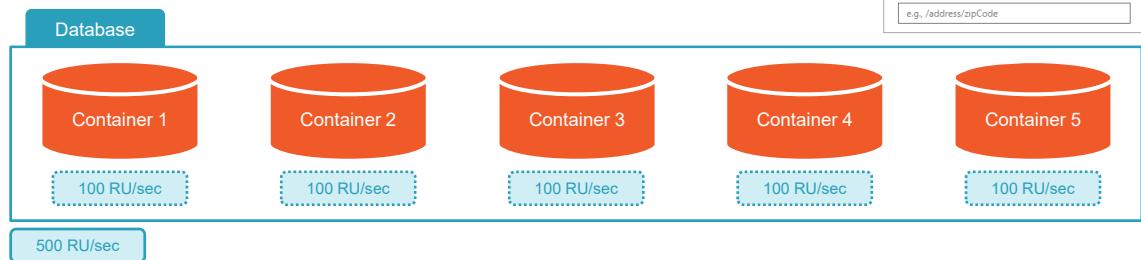
Database id (Create new selected)

Share throughput across containers (checked)

Database throughput (400 - unlimited RU/s)

Container id (e.g., Container1)

Partition key (e.g., /address/zipCode)



Provisioning Database Throughput

Migrating existing applications

May already be designed with separate containers per type

Differentiate solely on partition key

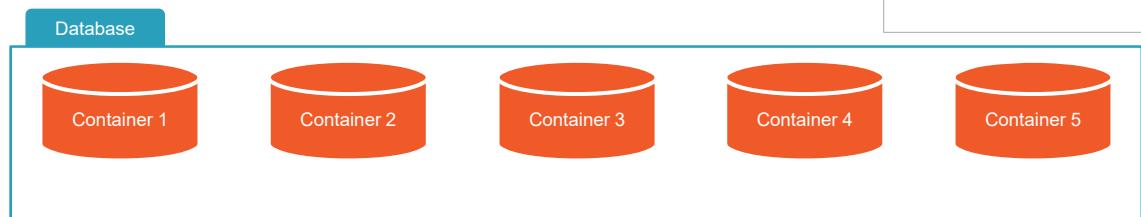
Containers share throughput needs, but different partitioning requirements

Database id (Use existing selected)

Container id (e.g., Container1)

Partition key (e.g., /address/zipCode)

Provision dedicated throughput for this container



Provisioning Database Throughput

Migrating existing applications

May already be designed with separate containers per type

Differentiate solely on partition key

Containers share throughput needs, but different partitioning requirements

Database id: Create new Use existing webstore

Container id: e.g., Container1

Partition key: e.g., /address/zipCode

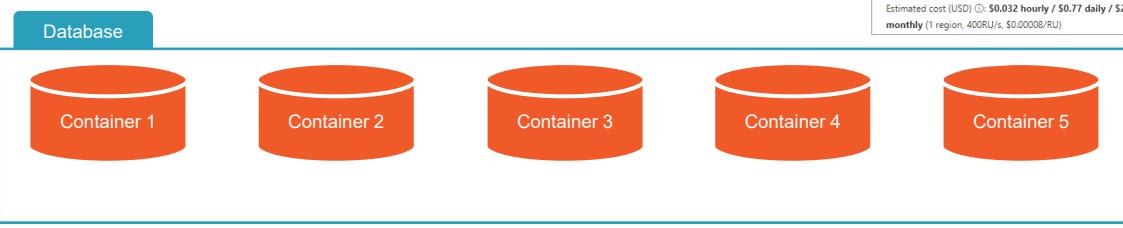
Provision dedicated throughput for this container*

Container throughput (400 - unlimited RU/s): Autoscale Manual

Estimated your required RU/s with capacity calculator.

400

Estimated cost (USD): \$0.032 hourly / \$0.77 daily / \$23.36 monthly (1 region, 400RU/s, \$0.00008/RU)



Provisioning Database Throughput

Migrating existing applications

May already be designed with separate containers per type

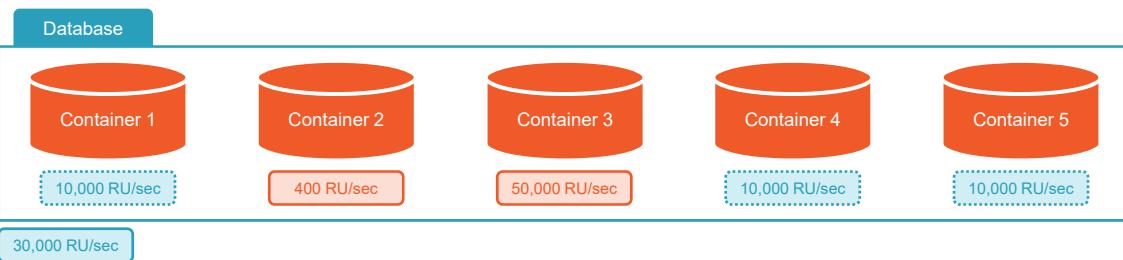
Differentiate solely on partition key

Containers share throughput needs, but different partitioning requirements

Mix and Match

Distribute database throughput across some containers

Provision other containers individually



Provisioning Database Throughput

Migrating existing applications

May already be designed with separate containers per type

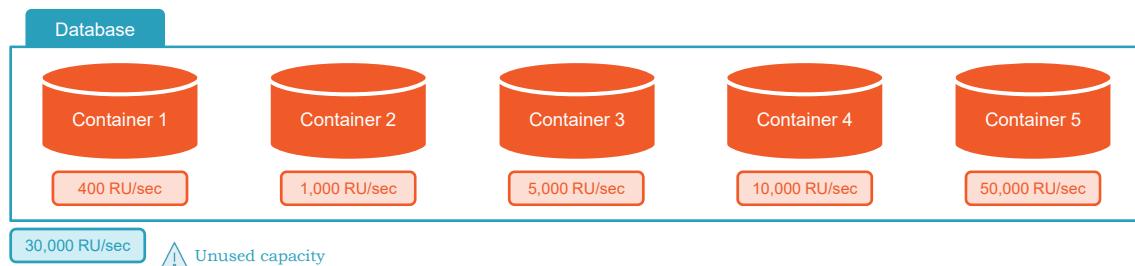
Differentiate solely on partition key

Containers share throughput needs, but different partitioning requirements

Mix and Match

Distribute database throughput across some containers

Provision other containers individually



Whiteboarding the Cost

Baseline

Read item	SQL Query	Write item
1 RU	~ 2.8 RUs	~ 10 RUs

Application checklist

- What does a typical item look like?
- What are the typical queries that users will run?
- How many writes per second are required?
- How many queries per second are required?
- What is the acceptable consistency level?
- What is the indexing policy?



Using the Capacity Calculator

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



Microsoft Azure | Azure Cosmos DB > Capacity Calculator

The calculator below offers you a quick estimate of the workload cost on Azure Cosmos DB. For a more precise estimate and ability to tweak more parameters, please sign in with an account you use for Azure.

Azure Cosmos DB Account Settings

The simplified Azure Cosmos DB calculator assumes commonly used settings for indexing policy, consistency, and other parameters. For a more accurate estimate, please sign in to provide your workload details.

API	SQL (Core)
Number of regions	1
Multi-region writes	<input checked="" type="radio"/> Disabled <input type="radio"/> Enabled

Workload per region

For a more accurate cost estimate based on your own data, please sign in and upload your sample data.

Total data stored in transactional store	10 GB
Use Analytical Store	<input checked="" type="radio"/> Off <input type="radio"/> On
Item size	1 KB
Point reads/sec	50
Creates/sec	10
Updates/sec	10
Deletes/sec	0
Queries/sec	1

Cost Estimate

Transactional Storage

Cost per GB/month	0.25 USD
Total Data stored per region	x 10 GB
EST. STORAGE COST PER MONTH	2.50 USD

Transactional Workload

Cost per 100 RU/s per hour	0.008 USD
EST. THROUGHPUT REQUIRED	Show Details x 400 RU/s
* Minimum throughput required is 400 RU/s	
EST. WORKLOAD COST/MONTH	23.36 USD

Number of regions

EST. TOTAL COST/MONTH	25.86 USD
-----------------------	-----------

Create Cosmos DB account

NEW TO AZURE COSMOS DB? CREATE A NEW FREE TIER ACCOUNT
Learn more

HAVE AN APP WITH BURSTY TRAFFIC? TRY OUT SERVERLESS
Learn more

MIGRATE TO AZURE COSMOS DB NOW
Learn more

Visual Studio Live! Austin 2022

Microsoft Azure | Azure Cosmos DB > Capacity Calculator

The calculator below offers you a quick estimate of the workload cost on Azure Cosmos DB. For a more precise estimate and ability to tweak more parameters, please sign in with an account you use for Azure.

Azure Cosmos DB Account Settings

The simplified Azure Cosmos DB calculator assumes commonly used settings for indexing policy, consistency, and other parameters. For a more accurate estimate, please sign in to provide your workload details.

API: SQL (Core)

Number of regions: 1

Multi-region writes: Disabled

Workload per region

Total data stored in transactional store: 10 GB

Use Analytical Store: Off

Item size: 1 KB

Point reads/sec: 1000

Creates/sec: 200

Updates/sec: 50

Deletes/sec: 0

Queries/sec: 0

Calculate

Settings have been updated. Please calculate again.

Cost Estimate

Transactional Storage

Cost per GB/month	0.25 USD
Total Data stored per region	x 10 GB
EST. STORAGE COST PER MONTH	2.50 USD

Transactional Workload

Cost per 100 RU/s per hour	0.008 USD
EST. THROUGHPUT REQUIRED	Show Details x 400 RU/s
* Minimum throughput required is 400 RU/s	
EST. WORKLOAD COST/MONTH	23.36 USD

Number of regions: x 1

EST. TOTAL COST/MONTH: 25.86 USD

Create Cosmos DB account

NEW TO AZURE COSMOS DB? CREATE A NEW FREE TIER ACCOUNT
[Learn more](#)

HAVE AN APP WITH BURSTY TRAFFIC? TRY OUT SERVERLESS
[Learn more](#)

MIGRATE TO AZURE COSMOS DB NOW
[Learn more](#)

Microsoft Azure | Azure Cosmos DB > Capacity Calculator

The calculator below offers you a quick estimate of the workload cost on Azure Cosmos DB. For a more precise estimate and ability to tweak more parameters, please sign in with an account you use for Azure.

Azure Cosmos DB Account Settings

The simplified Azure Cosmos DB calculator assumes commonly used settings for indexing policy, consistency, and other parameters. For a more accurate estimate, please sign in to provide your workload details.

API: SQL (Core)

Number of regions: 1

Multi-region writes: Disabled

Workload per region

Total data stored in transactional store: 10 GB

Use Analytical Store: Off

Item size: 1 KB

Point reads/sec: 1000

Creates/sec: 200

Updates/sec: 50

Deletes/sec: 0

Queries/sec: 0

Calculate

Cost Estimate

Transactional Storage

Cost per GB/month	0.25 USD
Total Data stored per region	x 10 GB
EST. STORAGE COST PER MONTH	2.50 USD

Transactional Workload

Cost per 100 RU/s per hour	0.008 USD
EST. THROUGHPUT REQUIRED	Show Details x 2,485 RU/s
EST. WORKLOAD COST/MONTH	145.14 USD

Number of regions: x 1

EST. TOTAL COST/MONTH: 147.64 USD

Create Cosmos DB account

NEW TO AZURE COSMOS DB? CREATE A NEW FREE TIER ACCOUNT
[Learn more](#)

HAVE AN APP WITH BURSTY TRAFFIC? TRY OUT SERVERLESS
[Learn more](#)

MIGRATE TO AZURE COSMOS DB NOW
[Learn more](#)

Microsoft Azure | Azure Cosmos DB > Capacity Calculator

The calculator below offers you a quick estimate of the workload cost on Azure Cosmos DB. For a more precise estimate and ability to tweak more parameters, please [sign in](#) to provide your workload details.

Azure Cosmos DB Account Settings

The simplified Azure Cosmos DB calculator assumes commonly used settings for indexing policy, consistency, and other parameters. For a more accurate estimate, please [sign in](#) to provide your workload details.

API: SQL (Core)

Number of regions: 1

Multi-region writes: Disabled

Workload per region

Total data stored in transactional store: 10 GB

Use Analytical Store: Off

Item size: 1 KB

Point reads/sec: 1000

Creates/sec: 200

Updates/sec: 50

Deletes/sec: 0

Queries/sec: 0

Cost Estimate

Transactional Storage

Cost per GB/month	0.25 USD
Total Data stored per region	x 10 GB
EST. STORAGE COST PER MONTH	2.50 USD

Transactional Workload

Cost per 100 RU/s per hour	0.008 USD
EST. THROUGHPUT REQUIRED Hide Details	x 2,485 RU/s
Throughput for Queries	0 RU/s
Throughput for Point Reads	1,000 RU/s
Throughput for Creates	990 RU/s
Throughput for Updates	495 RU/s
Throughput for Deletes	0 RU/s
EST. WORKLOAD COST/MONTH	145.14 USD

Number of regions: 1

EST. TOTAL COST/MONTH: 147.64 USD

Create Cosmos DB account

NEW TO AZURE COSMOS DB? CREATE A NEW FREE TIER ACCOUNT
[Learn more](#)

HAVE AN APP WITH BURSTY TRAFFIC? TRY OUT SERVERLESS
[Learn more](#)

Calculate

Pricing

Storage	Consumption-based SSD storage \$0.25 for 1 GB per month per region	<table border="1"> <thead> <tr><th></th><th>A</th><th>B</th><th>C</th></tr> </thead> <tbody> <tr><td>1</td><td>1 GB</td><td>10 GB</td><td></td></tr> <tr><td>2</td><td>\$ 0.25</td><td>\$ 2.50</td><td>/month</td></tr> </tbody> </table>		A	B	C	1	1 GB	10 GB		2	\$ 0.25	\$ 2.50	/month								
	A	B	C																			
1	1 GB	10 GB																				
2	\$ 0.25	\$ 2.50	/month																			
Provisioned throughput	Manual provisioned throughput \$0.008/hr for 100 RU/sec per region Minimum 400 RU/sec	<table border="1"> <thead> <tr><th></th><th>A</th><th>B</th><th>C</th></tr> </thead> <tbody> <tr><td>1</td><td>100 RU/s</td><td>400 RU/s</td><td></td></tr> <tr><td>2</td><td>\$ 0.008</td><td>\$ 0.032</td><td>/hour</td></tr> <tr><td>3</td><td>\$ 0.192</td><td>\$ 0.768</td><td>/day</td></tr> <tr><td>4</td><td>\$ 5.856</td><td>\$ 23.424</td><td>/month</td></tr> </tbody> </table>		A	B	C	1	100 RU/s	400 RU/s		2	\$ 0.008	\$ 0.032	/hour	3	\$ 0.192	\$ 0.768	/day	4	\$ 5.856	\$ 23.424	/month
	A	B	C																			
1	100 RU/s	400 RU/s																				
2	\$ 0.008	\$ 0.032	/hour																			
3	\$ 0.192	\$ 0.768	/day																			
4	\$ 5.856	\$ 23.424	/month																			
Autoscale provisioned throughput \$0.012/hr for 100 RU/sec per region Minimum 400 RU/sec	<table border="1"> <thead> <tr><th></th><th>A</th><th>B</th><th>C</th></tr> </thead> <tbody> <tr><td>1</td><td>100 RU/s</td><td>400 RU/s</td><td></td></tr> <tr><td>2</td><td>\$ 0.012</td><td>\$ 0.048</td><td>/hour</td></tr> <tr><td>3</td><td>\$ 0.288</td><td>\$ 1.152</td><td>/day</td></tr> <tr><td>4</td><td>\$ 8.784</td><td>\$ 35.136</td><td>/month</td></tr> </tbody> </table>		A	B	C	1	100 RU/s	400 RU/s		2	\$ 0.012	\$ 0.048	/hour	3	\$ 0.288	\$ 1.152	/day	4	\$ 8.784	\$ 35.136	/month	
	A	B	C																			
1	100 RU/s	400 RU/s																				
2	\$ 0.012	\$ 0.048	/hour																			
3	\$ 0.288	\$ 1.152	/day																			
4	\$ 8.784	\$ 35.136	/month																			
Serverless	Consumption-based per request \$0.25 per 1 million RUs	<table border="1"> <thead> <tr><th></th><th>A</th><th>B</th><th>C</th></tr> </thead> <tbody> <tr><td>1</td><td>1 M RUs</td><td>5 M RUs</td><td></td></tr> <tr><td>2</td><td>\$ 0.25</td><td>\$ 1.25</td><td>/month</td></tr> </tbody> </table>		A	B	C	1	1 M RUs	5 M RUs		2	\$ 0.25	\$ 1.25	/month								
	A	B	C																			
1	1 M RUs	5 M RUs																				
2	\$ 0.25	\$ 1.25	/month																			
More options	Reserved capacity, multi-master, analytic storage https://azure.microsoft.com/pricing/details/cosmos-db																					



Introduction to Azure Cosmos DB

Horizontal Partitioning



Achieving Elastic Scale

What is Partitioning?

Massive scale-out within a container



Containers

Single logical resource composed of multiple physical partitions



Partitions

Physical fixed-capacity data buckets

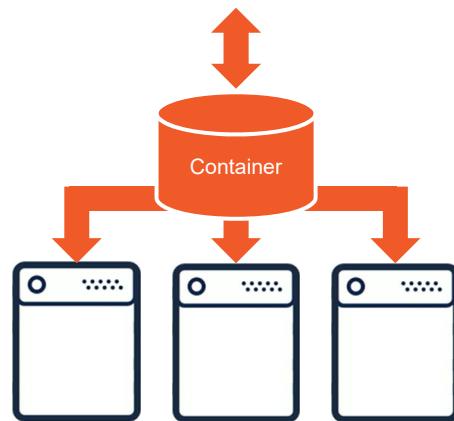


Automated Scale-Out

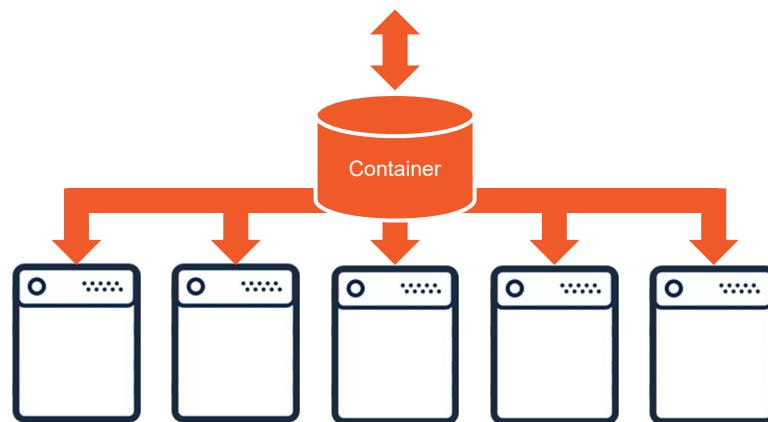
Cosmos DB transparently splits partitions to manage growth



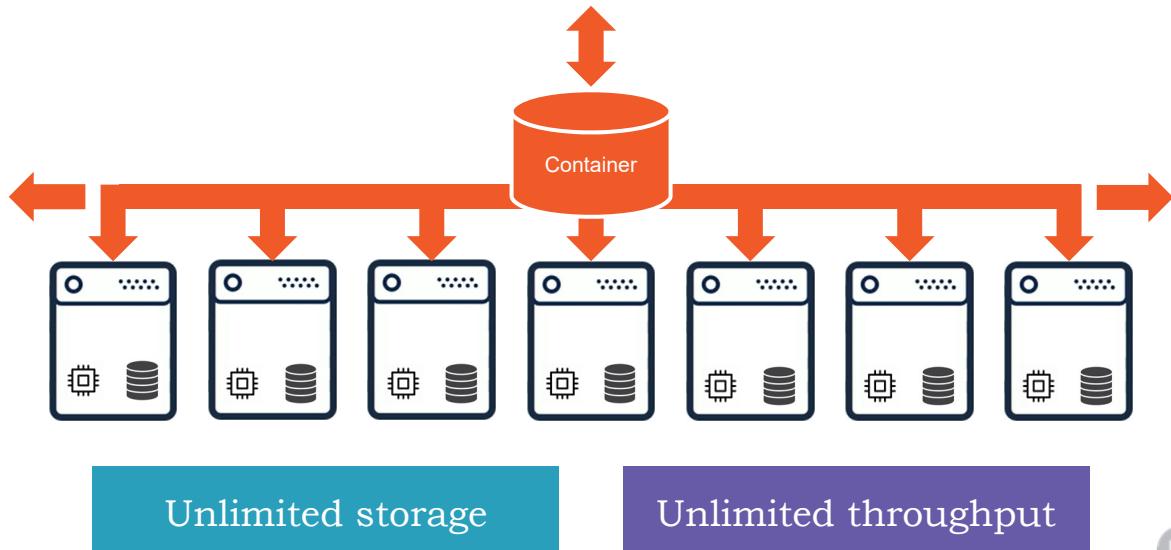
Achieving Elastic Scale



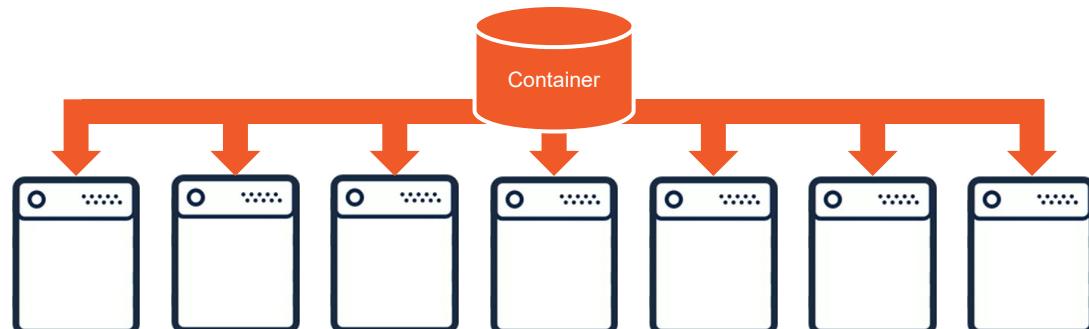
Achieving Elastic Scale



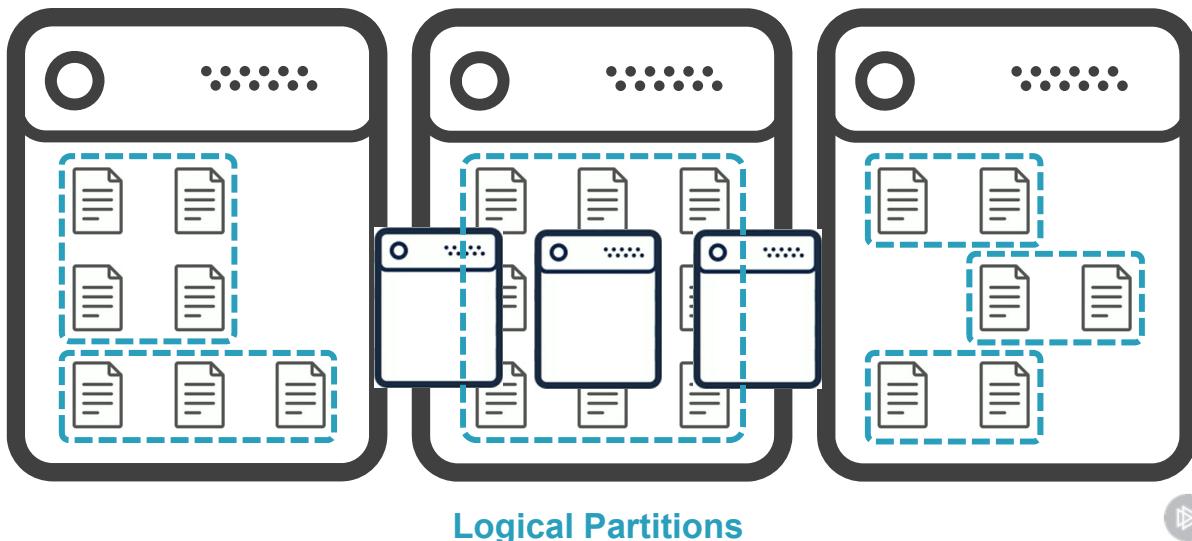
Achieving Elastic Scale



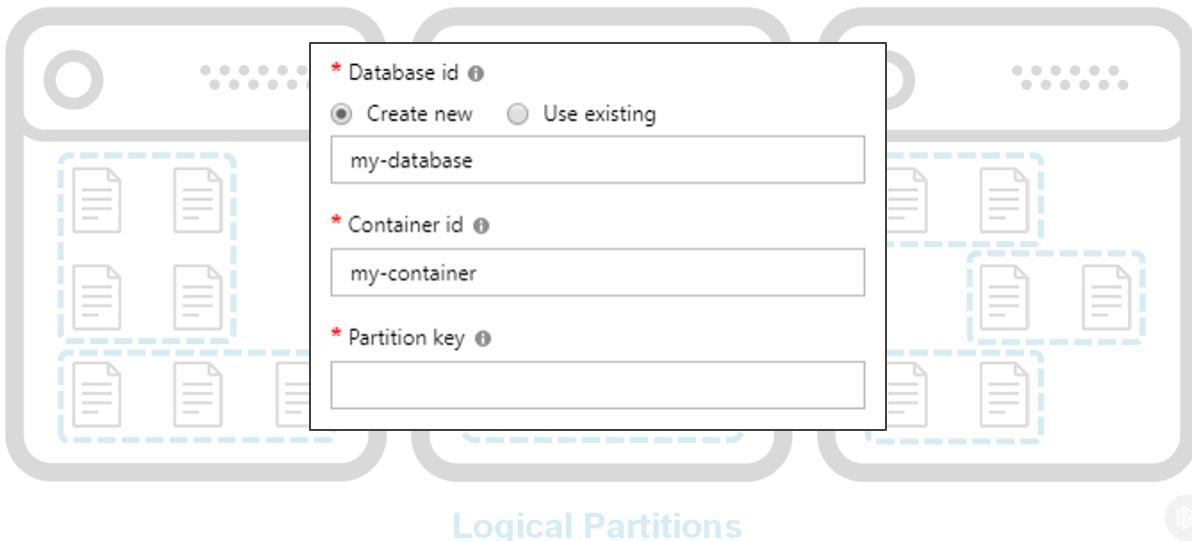
Understanding Logical Partitions



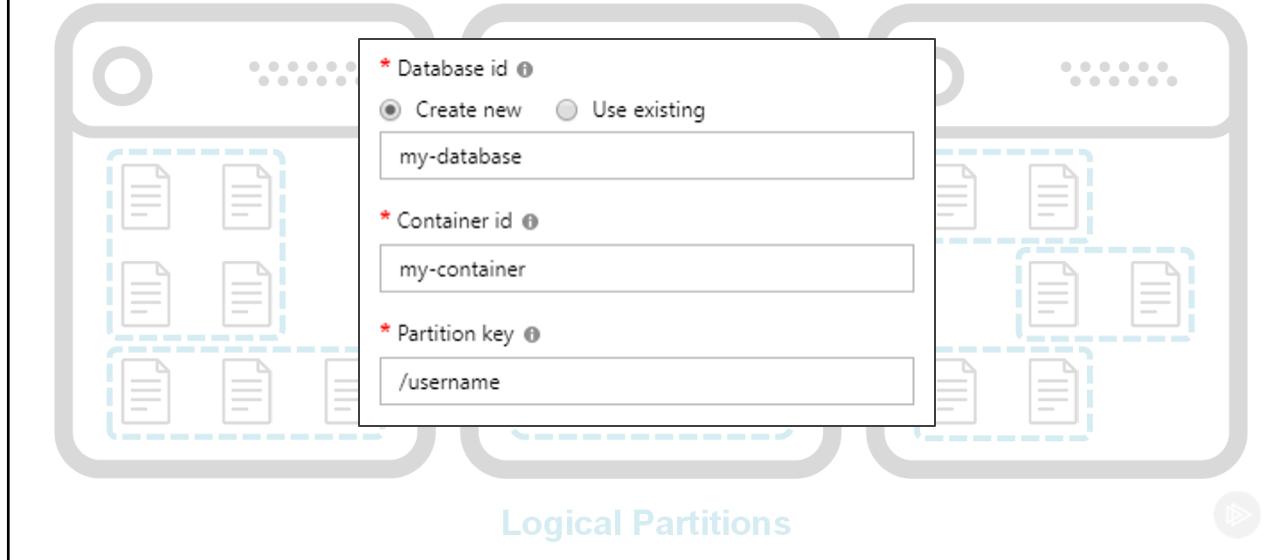
Understanding Logical Partitions



Understanding Logical Partitions



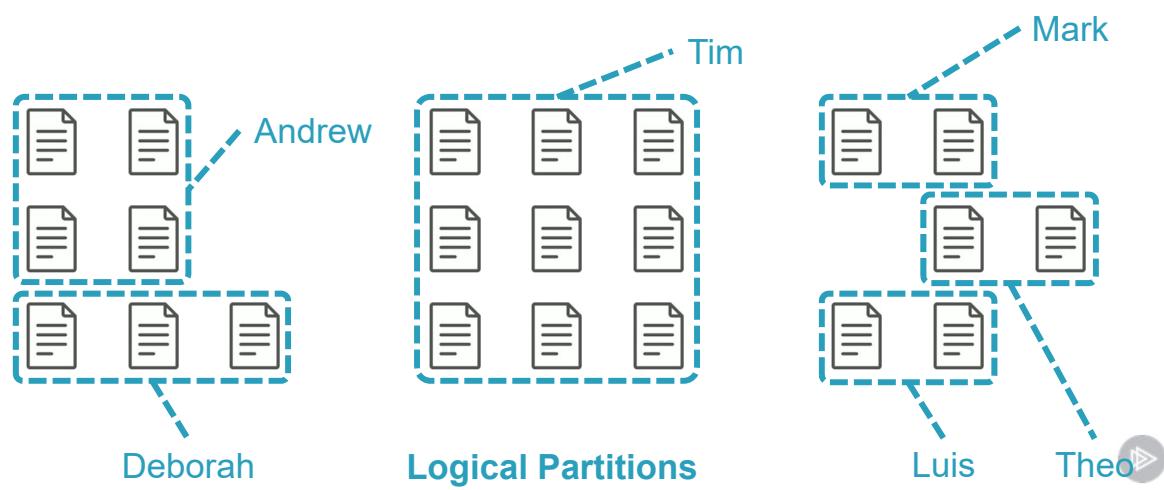
Understanding Logical Partitions



Logical Partitions



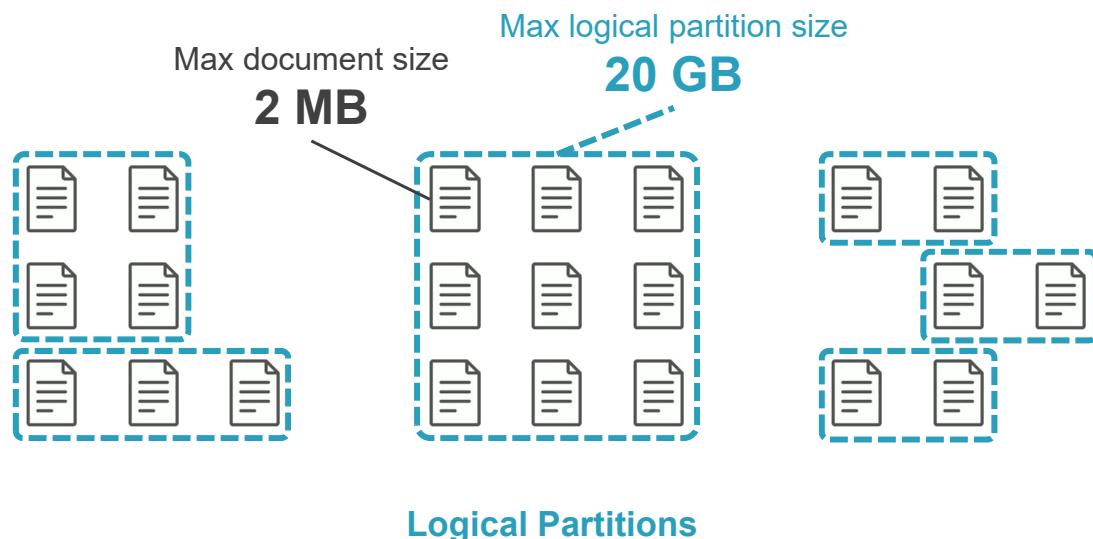
Understanding Logical Partitions



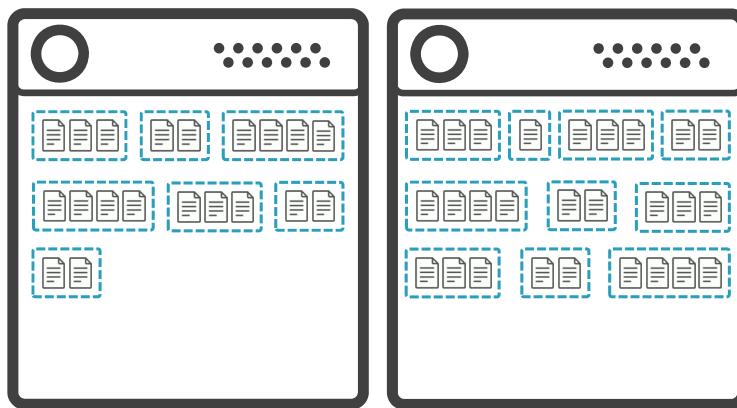
Logical Partitions



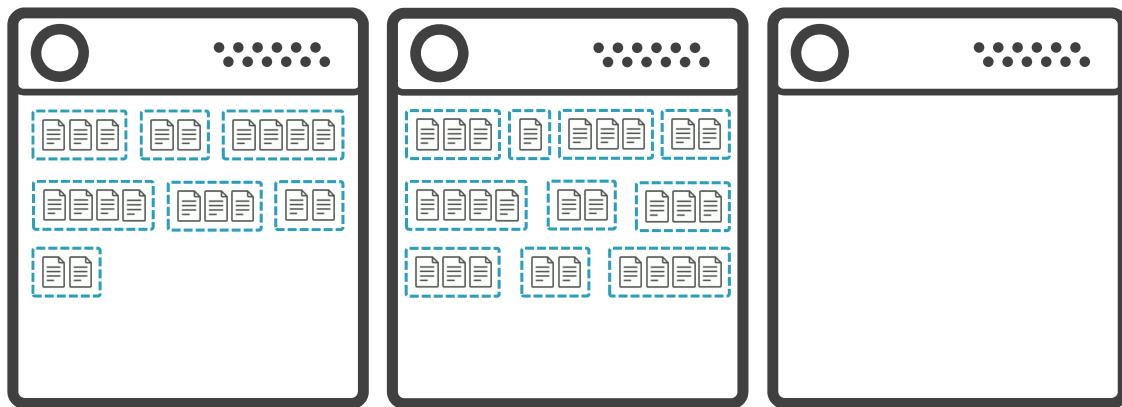
Understanding Logical Partitions



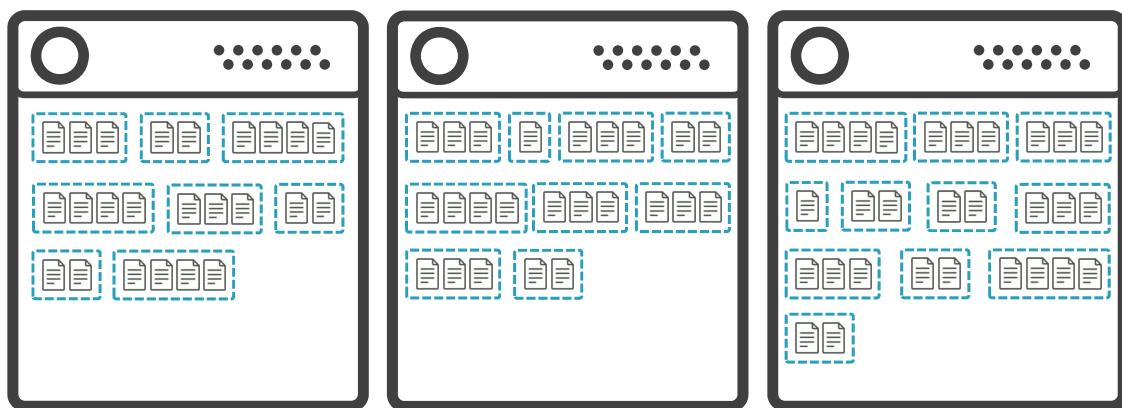
Partition Splits



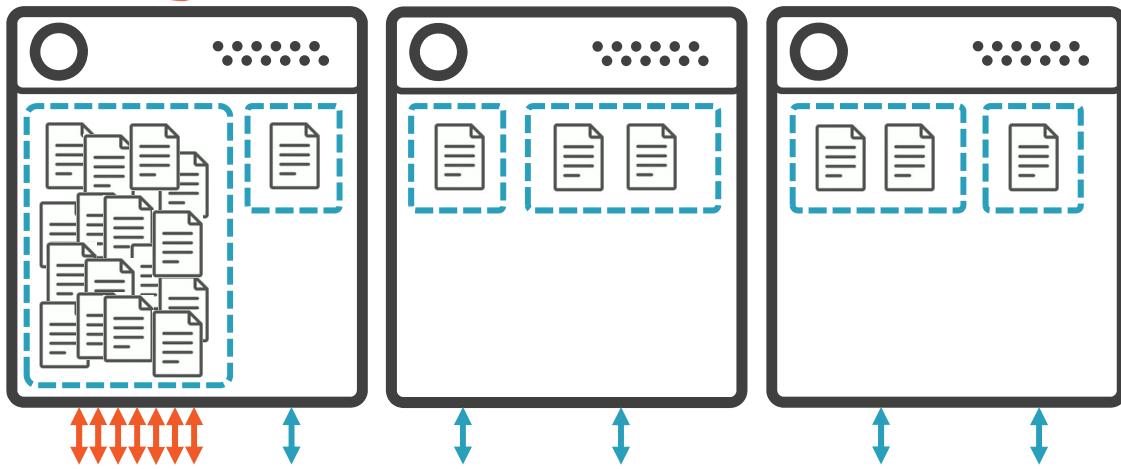
Partition Splits



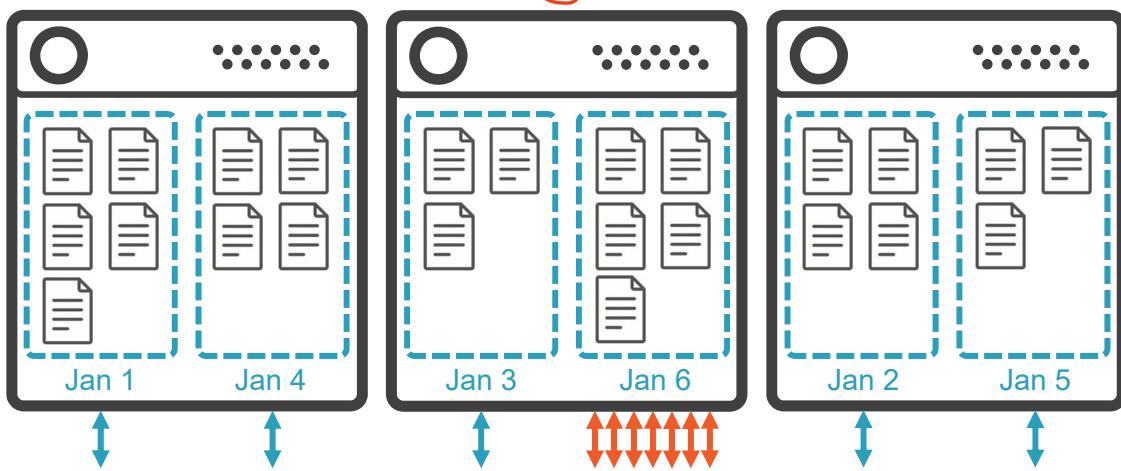
Partition Splits



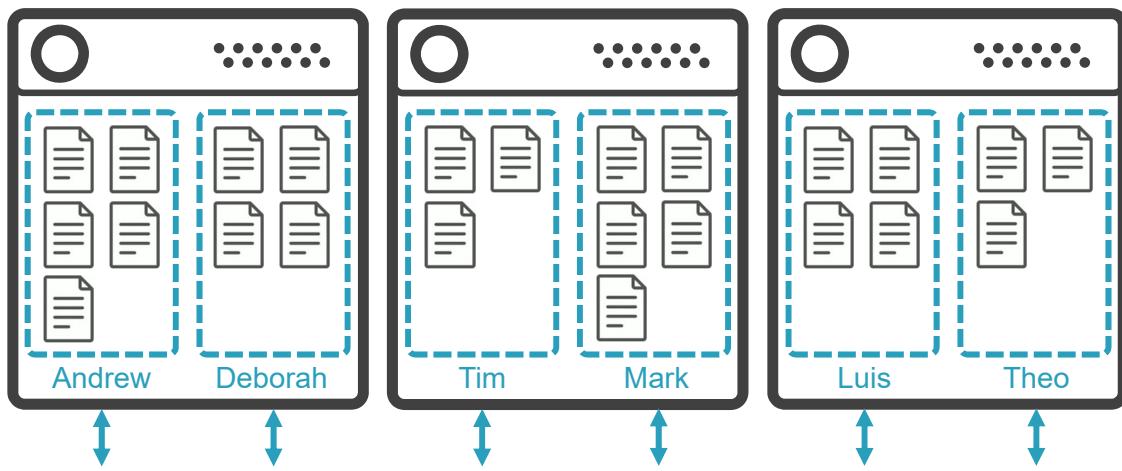
Avoiding Hot Partitions



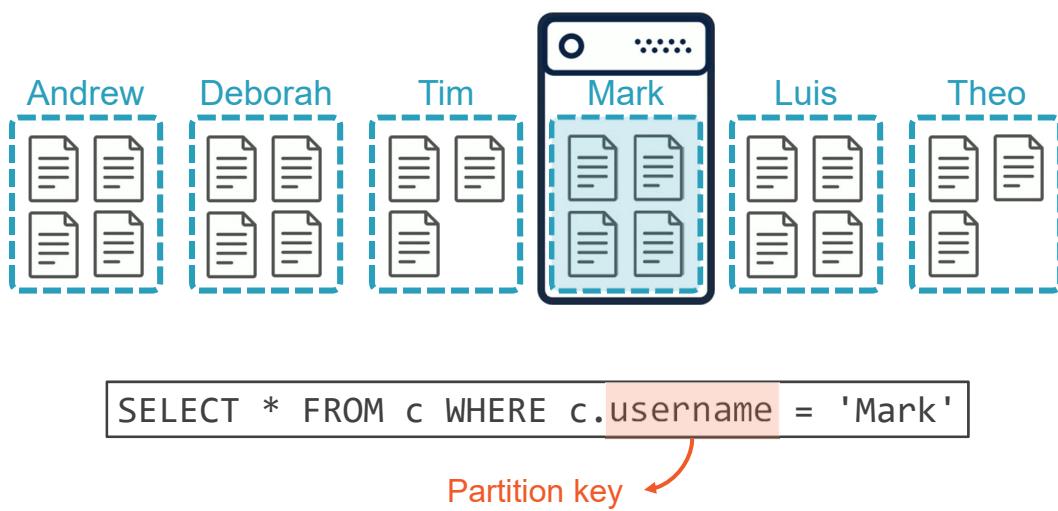
Avoiding Hot Partitions



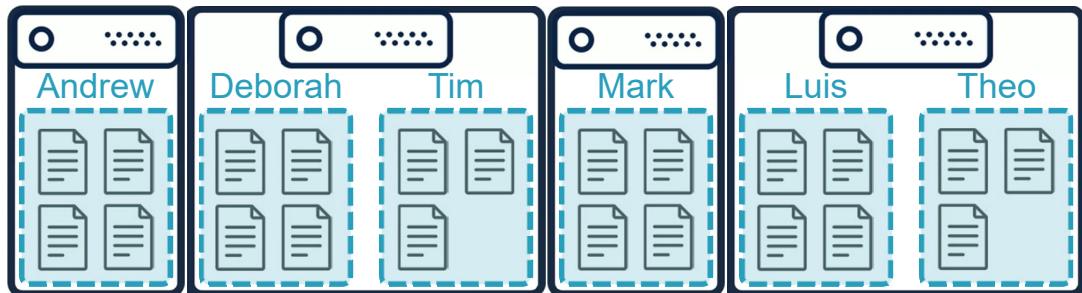
Avoiding Hot Partitions



Cross-partition Queries



Cross-partition Queries



```
SELECT * FROM c WHERE c.favoriteColor = 'Red'
```



Choosing the Right Partition Key

Based on common usage patterns

The right choice will deliver massive scale

Avoid performance bottlenecks

Ensure uniform distribution of both storage and throughput

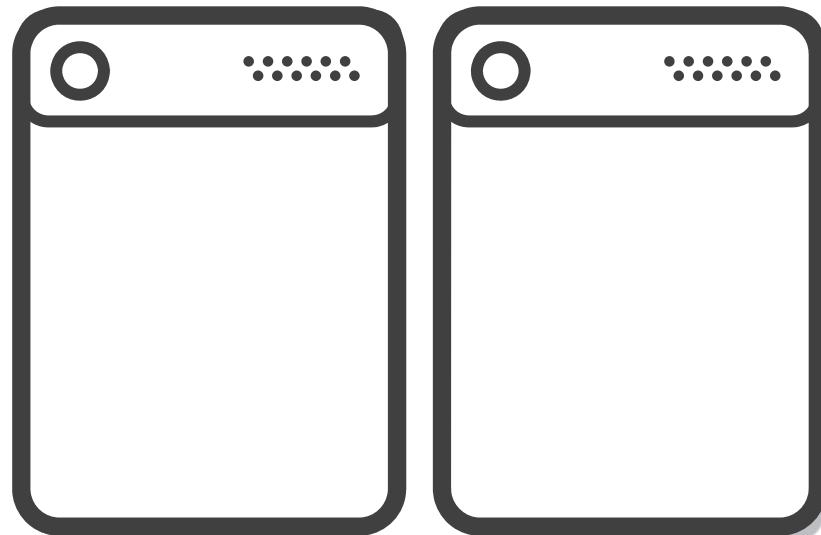
Boundary for queries and transactions

Minimize cross-partition queries

Stored procedures with ACID guarantees

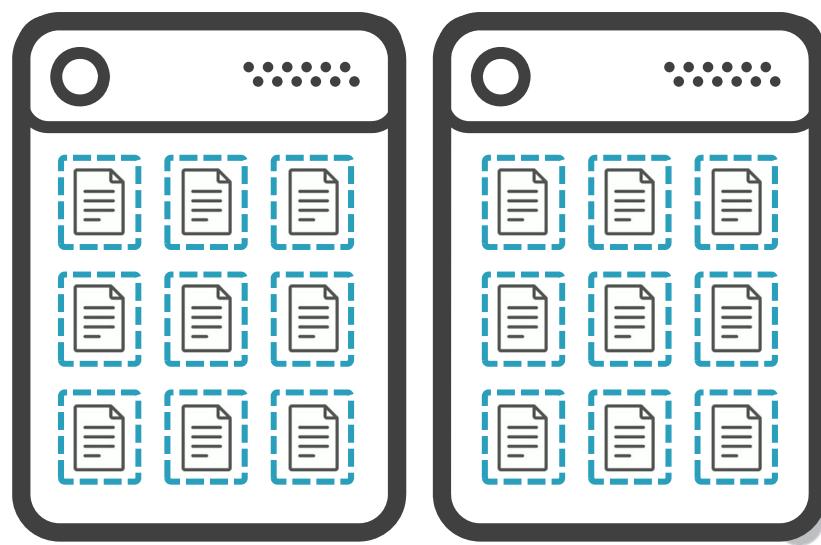


Common Partitioning Patterns



Common Partitioning Patterns

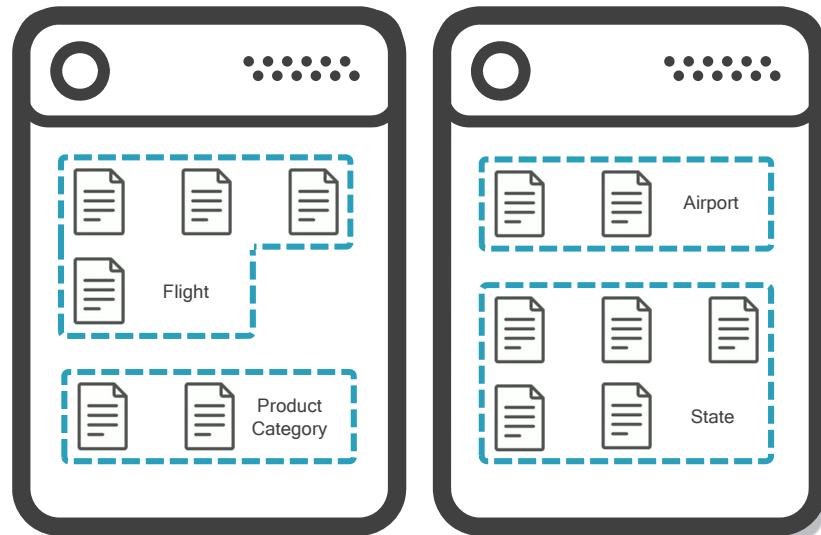
/id
Single-documents



Common Partitioning Patterns

/id
Single-documents

/type
Small lookup lists

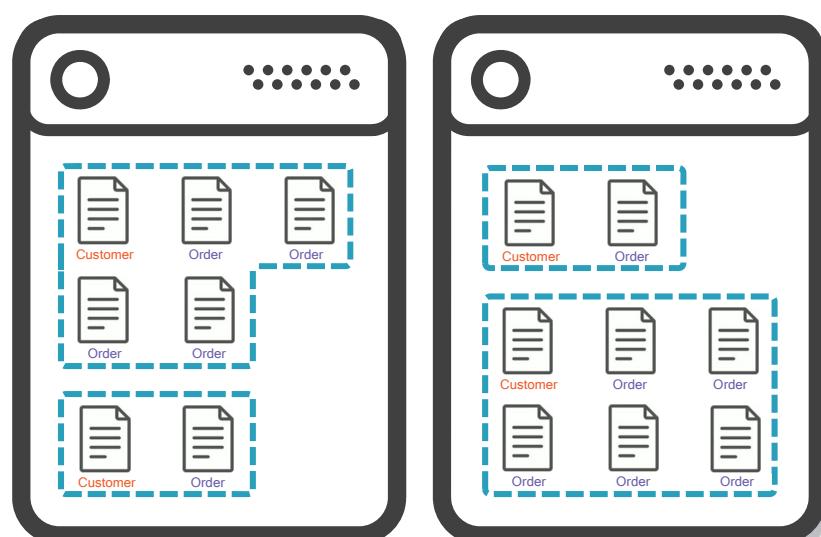


Common Partitioning Patterns

/id
Single-documents

/type
Small lookup lists

Other Example:
Optimize for queries /customerId



Common Partitioning Patterns

/id

Single-documents

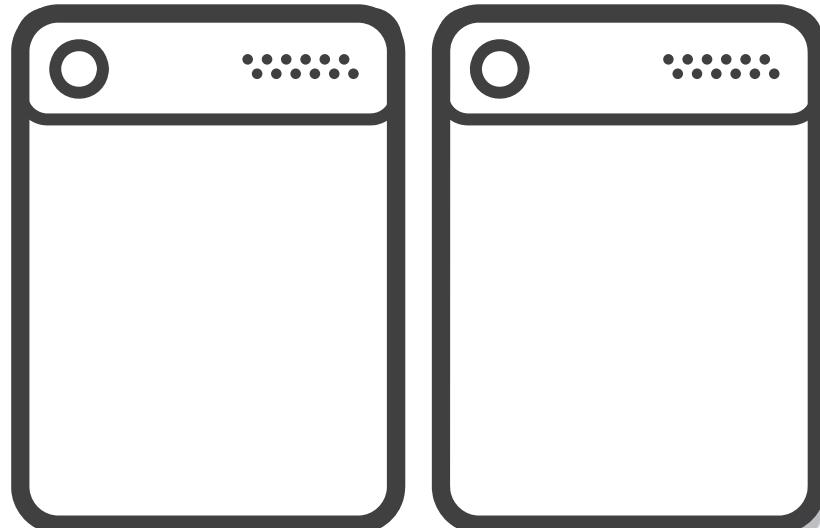
/type

Small lookup lists

Other

Example:
/customerId

Optimize for queries



Changing the Partition Key

Partition keys are immutable

Can't change for the container

Can't change for the document

Always use the same property path

For example:
/pk

Store a copy of the desired partition key

Always uniquely qualify the id property

Can be a GUID, but doesn't have to be

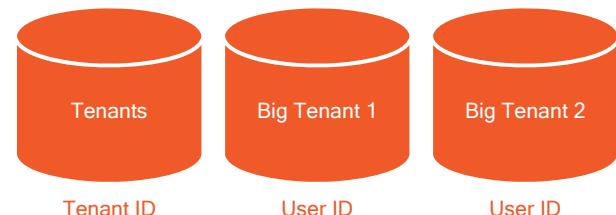
Avoid collisions during migration



Partition Granularity

Synthetic Partition Keys

IoT: Device ID
Multitenant: Tenant ID



Partition Granularity

Synthetic Partition Keys

IoT: Device ID
Multitenant: Tenant ID

Device ID + Month
Tenant ID + User ID

Hierarchical Partition Keys

Aka “subpartitioning”

Up to 3 levels

Leaf level limit = 20 GB
Parent level limit > 20 GB



Hierarchical Partition Keys

Currently in Private Preview

Signup:
<https://aka.ms/cosmos-subpartitioning-signup>



Hierarchical Partition Keys



Partition key: Tenant ID

```
SELECT * FROM c WHERE c.tenantId = 'TenantX'
```

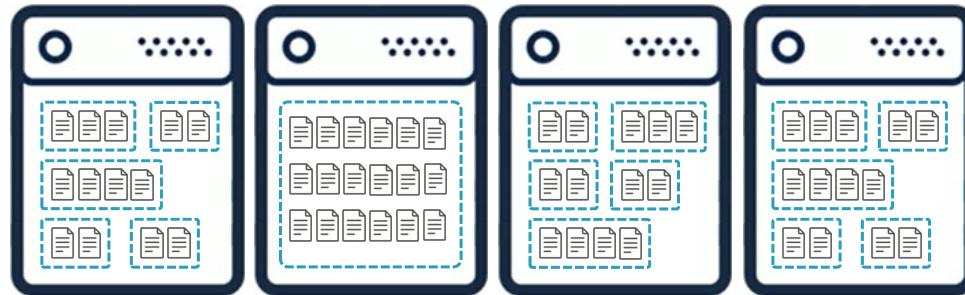
Single-partition

```
SELECT * FROM c WHERE c.tenantId = 'TenantX' AND c.userId = 'UserY'
```

Single-partition



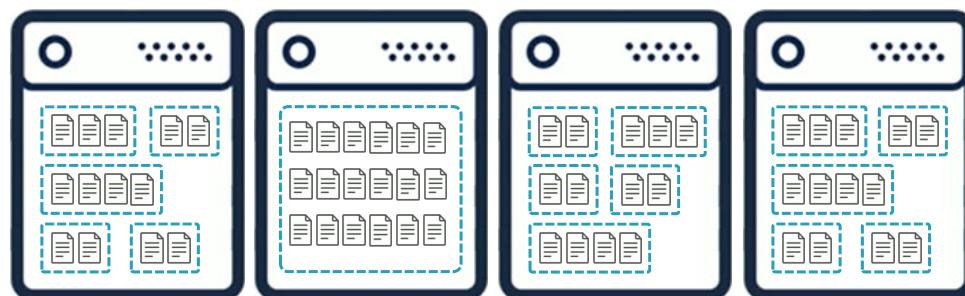
Hierarchical Partition Keys



Partition key: **Tenant ID**



Hierarchical Partition Keys



Synthetic partition key: **Tenant ID + User ID**

```
SELECT * FROM c WHERE c.tenantId = 'TenantX'
```

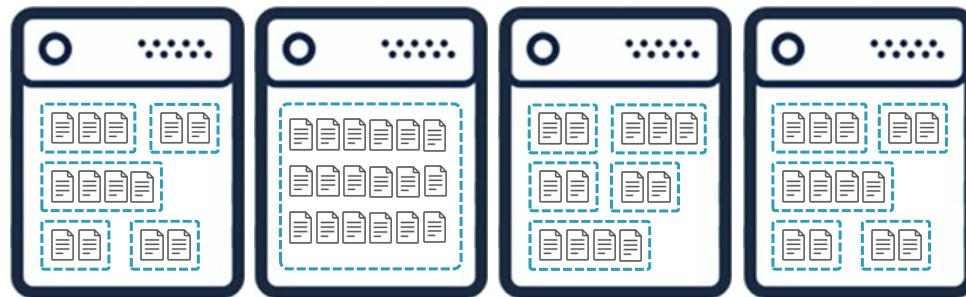
Cross-partition

```
SELECT * FROM c WHERE c.synthPk = 'TenantX_UserY'
```

Single-partition



Hierarchical Partition Keys

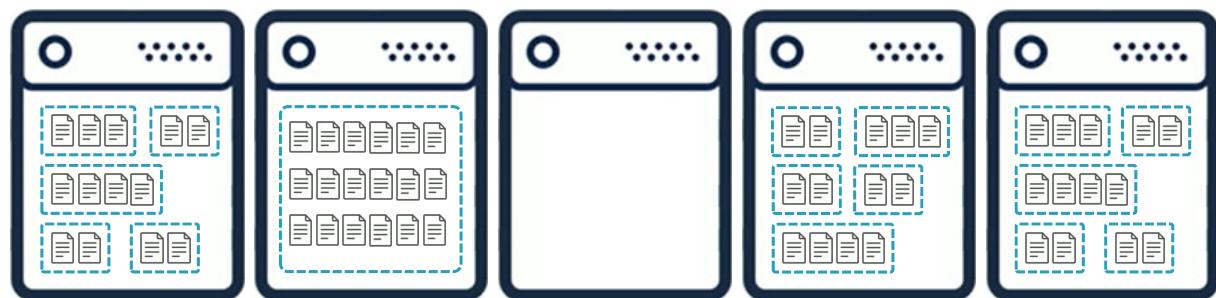


Hierarchical partition key: **Tenant ID / User ID**

Parent Level Leaf Level



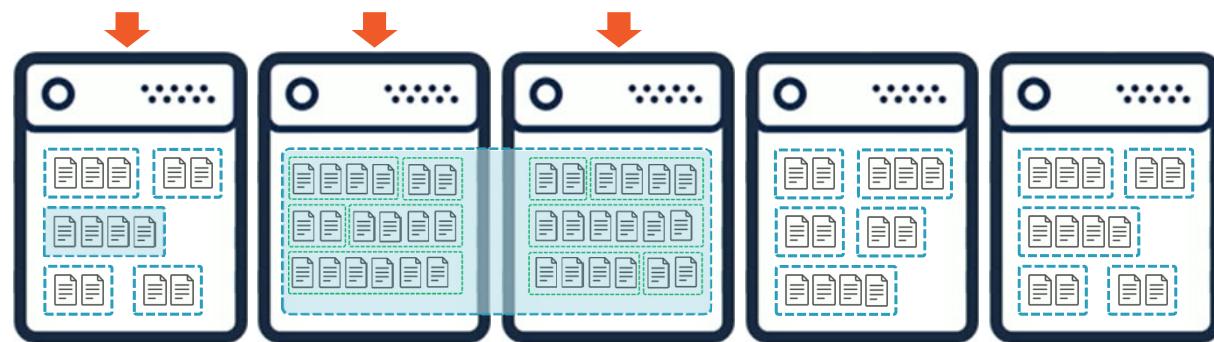
Hierarchical Partition Keys



Hierarchical partition key: **Tenant ID / User ID**



Hierarchical Partition Keys



Hierarchical partition key: Tenant ID / User ID

```
SELECT * FROM c WHERE c.tenantId = 'TenantX'
```

Sub-partition



Hierarchical Partition Keys



Hierarchical partition key: Tenant ID / User ID

```
SELECT * FROM c WHERE c.tenantId = 'TenantX'
```

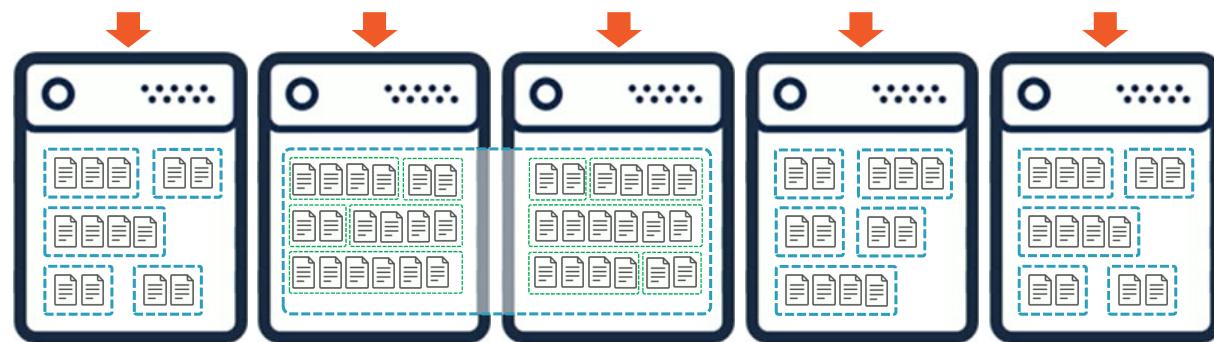
Sub-partition

```
SELECT * FROM c WHERE c.tenantId = 'TenantX' AND c.userId = 'UserY'
```

Single-partition



Hierarchical Partition Keys



Hierarchical partition key: Tenant ID / User ID

```
SELECT * FROM c WHERE c.tenantId = 'TenantX'
```

Sub-partition

```
SELECT * FROM c WHERE c.tenantId = 'TenantX' AND c.userId = 'UserY'
```

Single-partition

```
SELECT * FROM c WHERE c.userId = 'UserZ'
```

Cross-partition



Introduction to Azure Cosmos DB

Global Distribution



Replication – Why?

Performance

Multiple replicas
within a region

Ensures high availability
Across regions, brings data
closer to the consumer

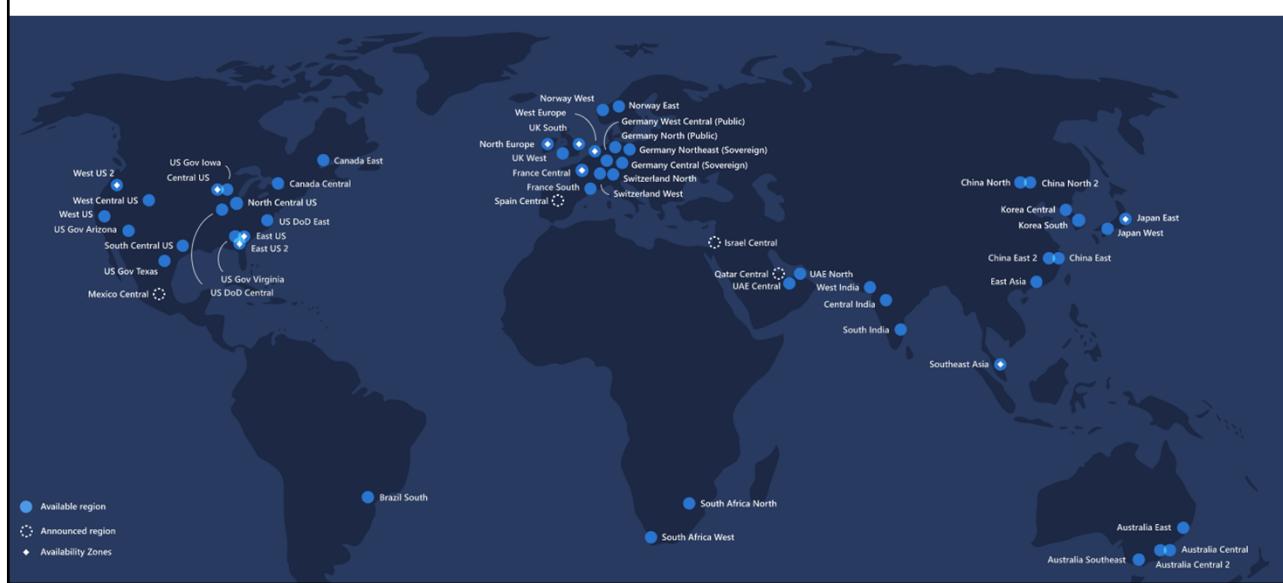
Business continuity

Replica failover
within a region

Globally, in the event of
major failure or natural
disaster



Azure Data Centers



Enabling Global Distribution

Associate multiple regions with your
Cosmos DB account
Limited to geo-fencing policies

Dynamically add
and remove regions
Associate (and disassociate) regions
with the click of a mouse

Multi-master
Enable writes across all regions,
with automatic failover

Save Discard Manual Failover Automatic Failover

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](#)

Enabling Global Distribution

Associate multiple regions with your
Cosmos DB account
Limited to geo-fencing policies

Dynamically add
and remove regions
Associate (and disassociate) regions
with the click of a mouse

Multi-master
Enable writes across all regions,
with automatic failover

Configure regions
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](#)

Write Region	Availability Zone	
East US	<input type="checkbox"/>	
Read Regions	Availability Zone	Action
West US	<input type="checkbox"/>	<input type="checkbox"/>
West Europe	<input type="checkbox"/>	<input type="checkbox"/>



Enabling Global Distribution

Associate multiple regions with your
Cosmos DB account
Limited to geo-fencing policies

Dynamically add
and remove regions
Associate (and disassociate) regions
with the click of a mouse

Multi-master
Enable writes across all regions,
with automatic failover

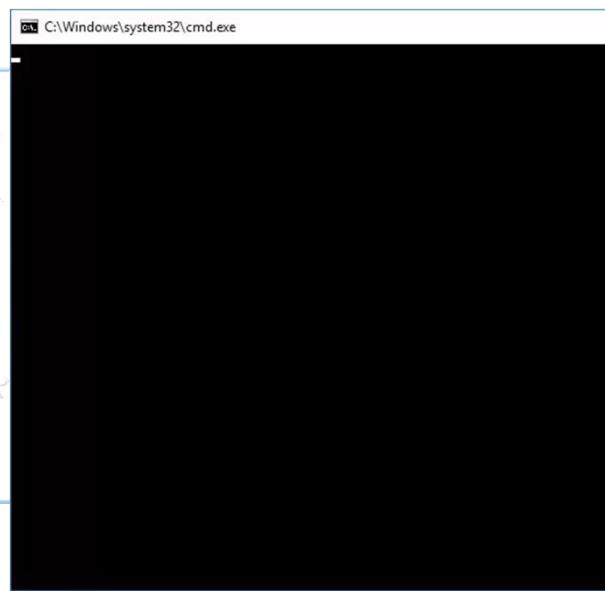
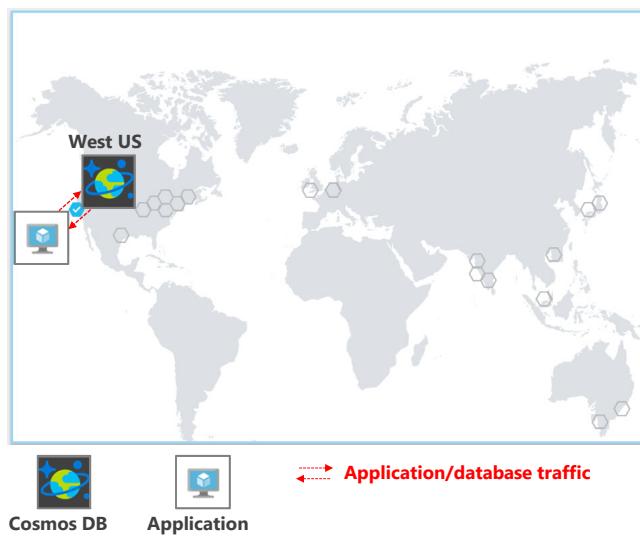
Configure regions
Multi-region writes ⓘ

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

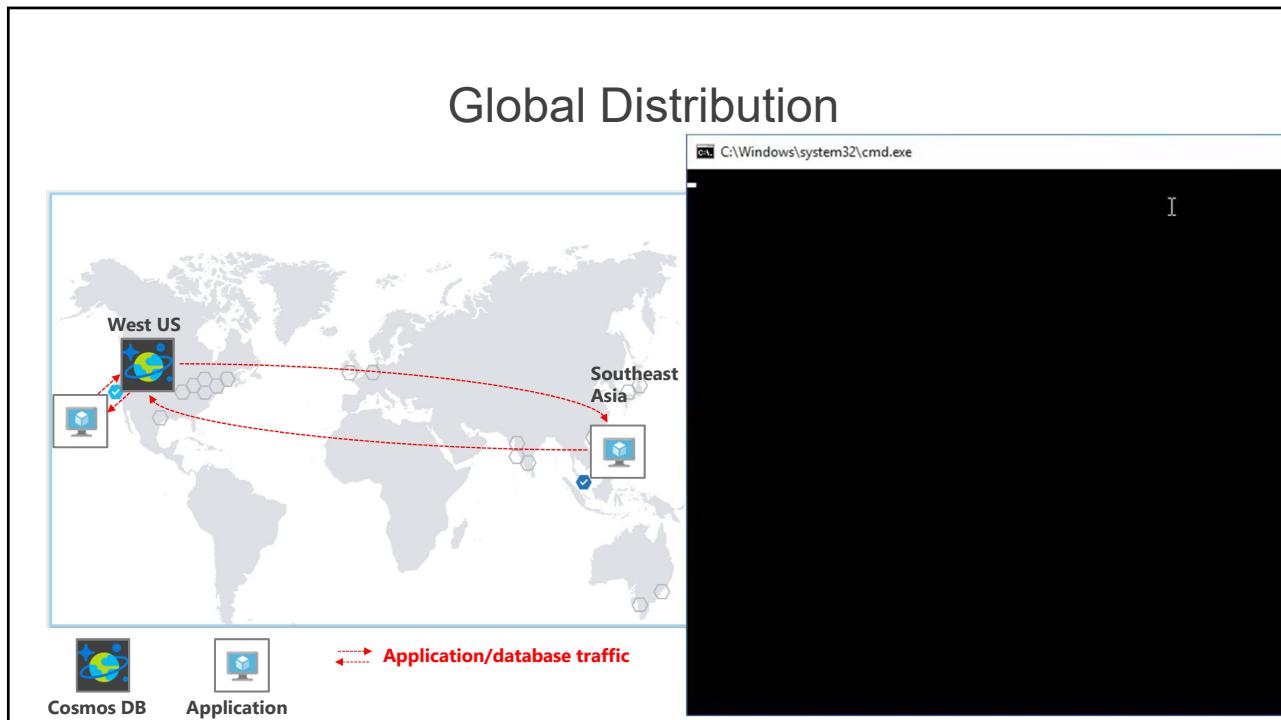
Regions	Reads Enabled	Writes Enabled	Availability Zone	Action
East US	✓	✓		<input type="button" value="Delete"/>
West US	✓	✓		<input type="button" value="Delete"/>
West Europe	✓	✓	<input type="checkbox"/>	<input type="button" value="Delete"/>



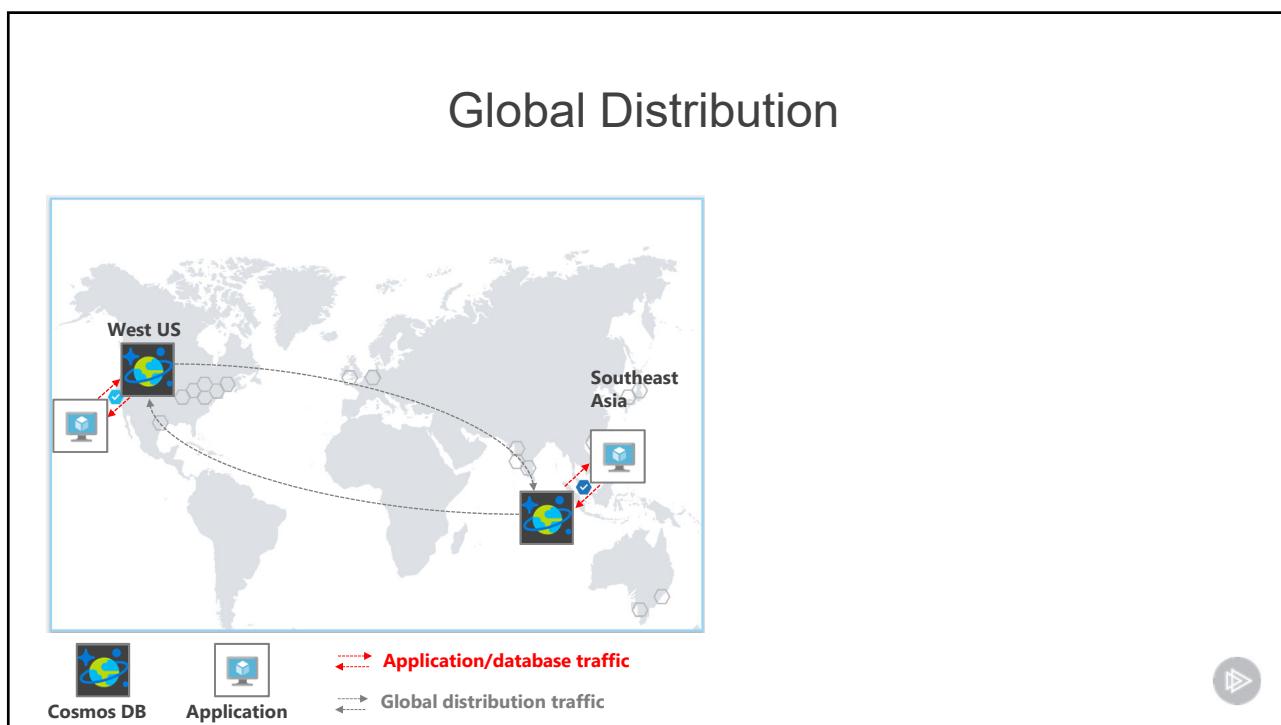
Global Distribution



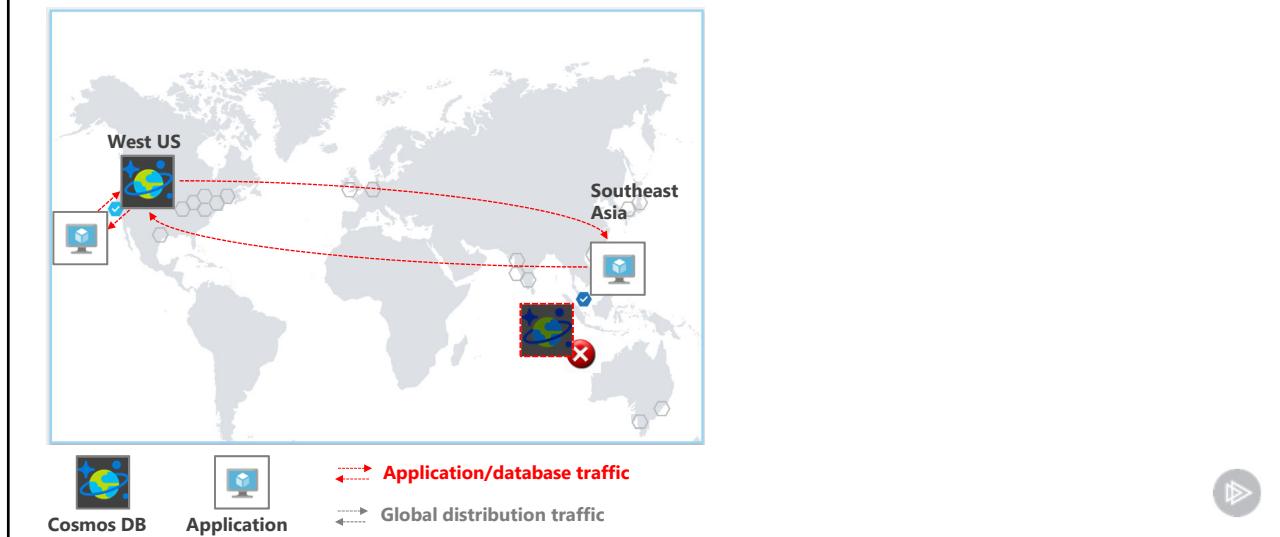
Global Distribution



I



Global Distribution



Multi-master Conflict Resolution

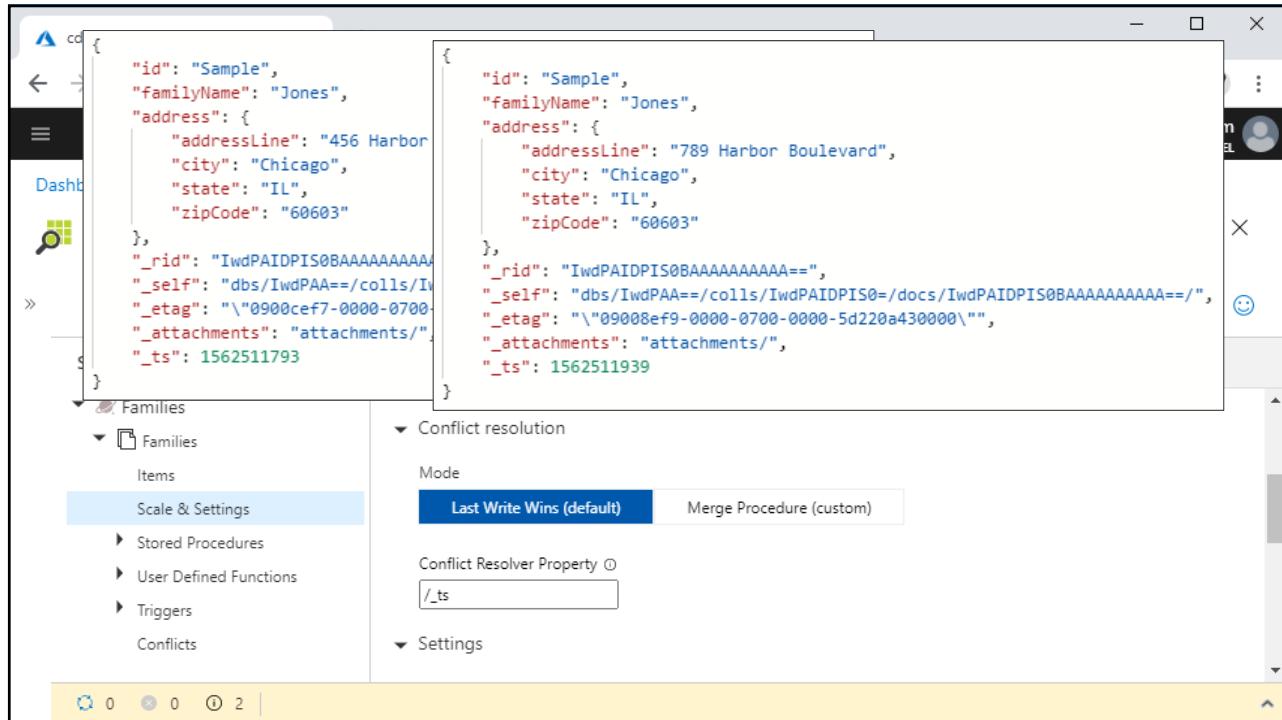
Every region is enabled for writes, inviting conflicts

Three options for conflict resolution

Last writer wins

Based on highest timestamp value in _ts





Multi-master Conflict Resolution

Every region is enabled for writes

Three options for conflict resolution

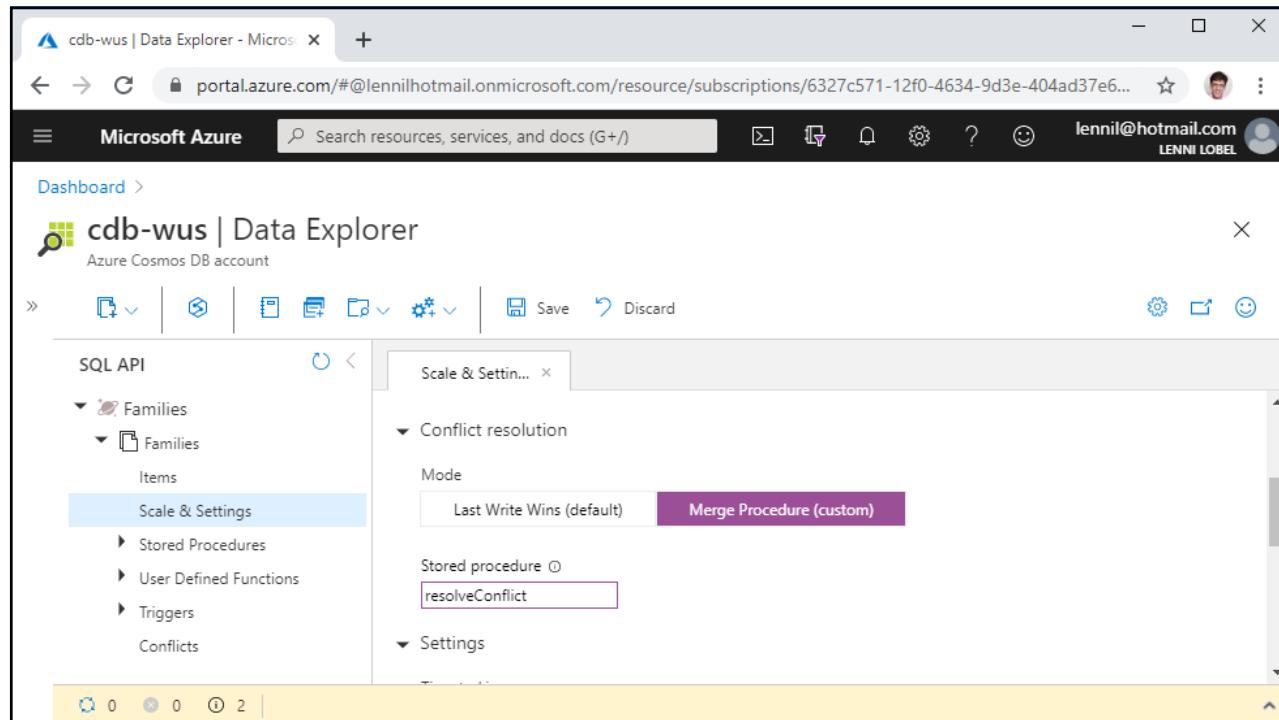
Last writer wins

Based on highest timestamp value in `_ts`

Custom procedure

Based on stored procedure result





Multi-master Conflict Resolution

Every region is enabled for writes

Three options for conflict resolution

Last writer wins

Based on highest timestamp value in _ts

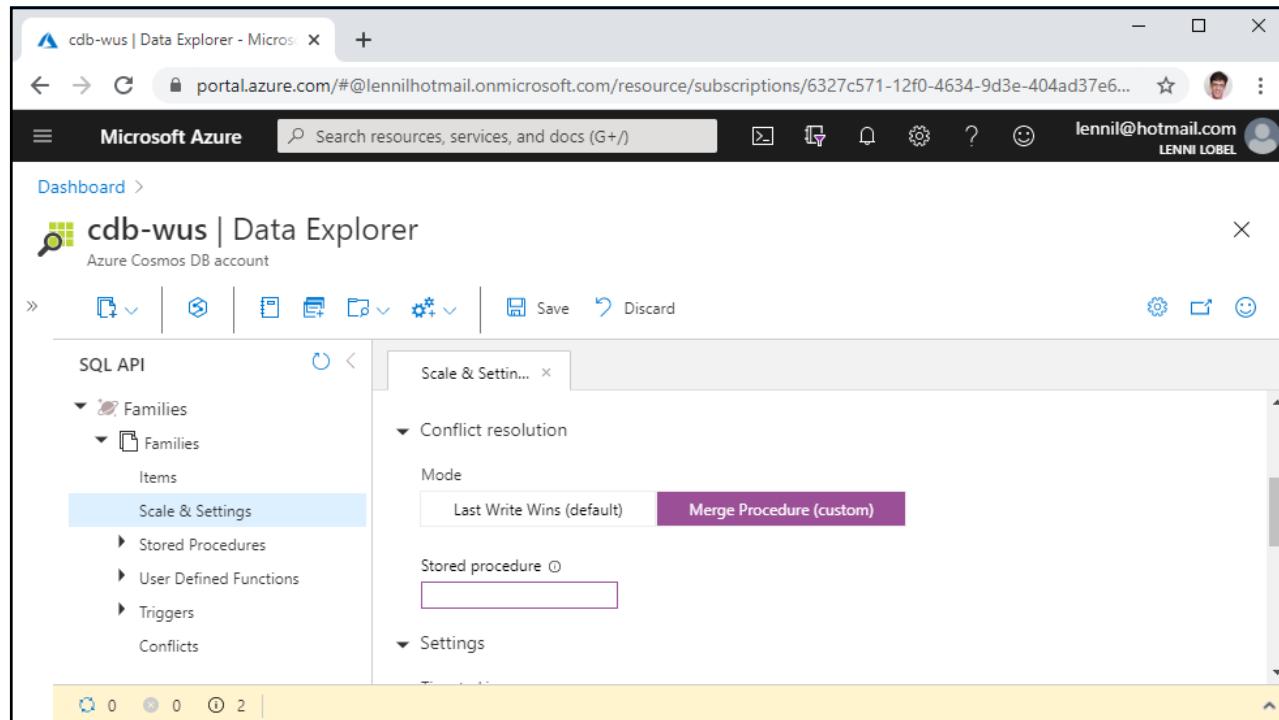
Custom procedure

Based on stored procedure result

Conflicts feed

Offline resolution





Replication and Consistency

Replication within a region

Multiple replicas invite “dirty reads”

Extremely rare, since replicas are updated extremely fast

Global replication

It takes hundreds of milliseconds to move data across continents

Much more common to experience dirty reads

How to ensure consistent reads across replicas?

Define a consistency level

Cleaner reads
Higher latency



Strong

Bounded
Staleness

Session

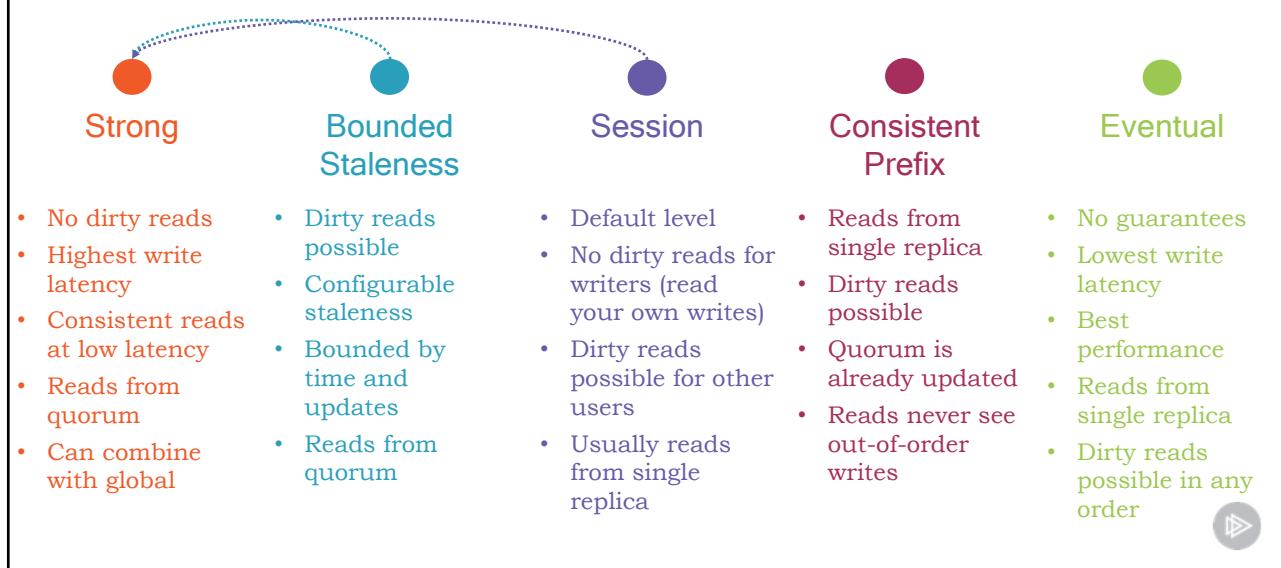
Consistent
Prefix

Eventual

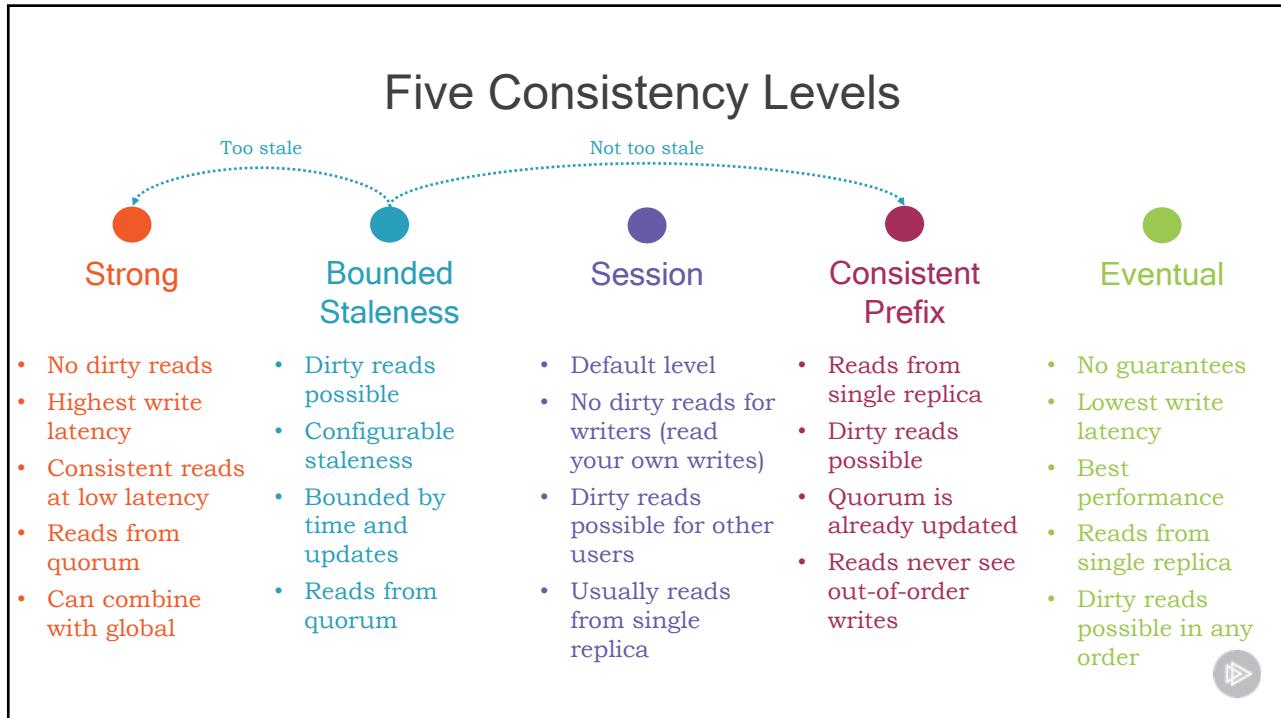
Dirtier reads
Lower latency



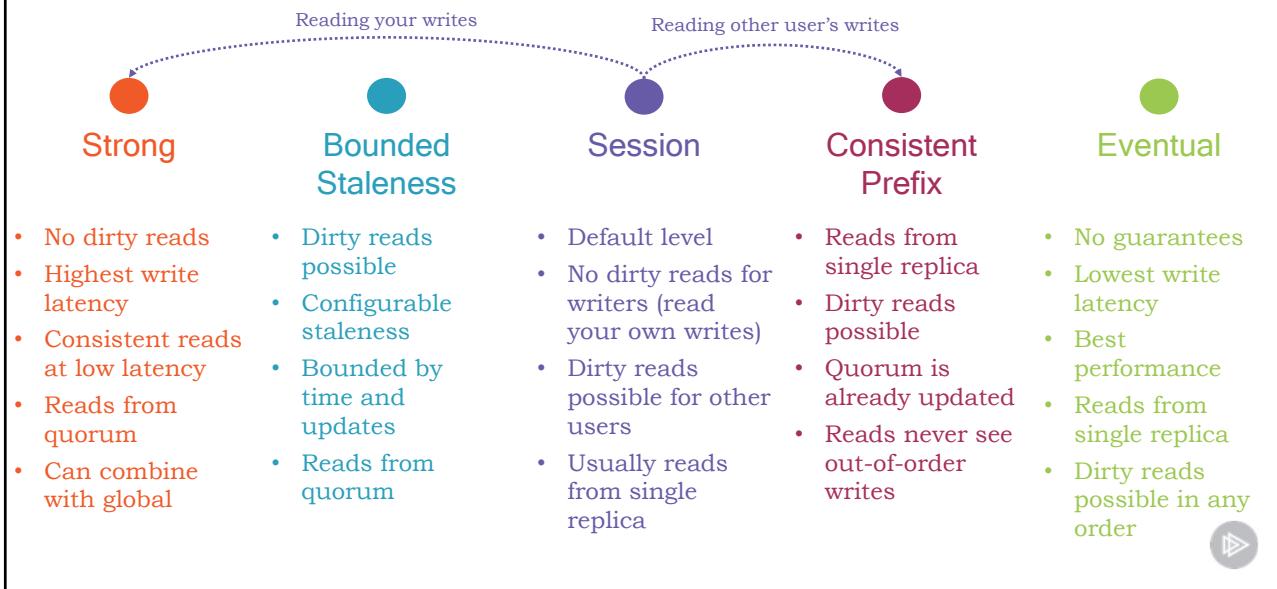
Five Consistency Levels



Five Consistency Levels



Five Consistency Levels



Setting the Consistency Level

Set default for entire account
Can be changed at any time



Microsoft Azure | Search resources, services, and docs (G+)

Dashboard > cdb-wus | Default consistency

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Quick start Notifications Data Explorer

Settings Features Replicate data globally Default consistency Firewall and virtual networks Private Endpoint Connections CORS Keys Add Azure Cognitive Search

STRONG BOUNDED STALENESS SESSION CONSISTENT PREFIX EVENTUAL

i 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.

Microsoft Azure | Search resources, services, and docs (G+)

Dashboard > cdb-wus | Default consistency

Overview Activity log Access control (IAM) Tags Diagnose and solve problems Quick start Notifications Data Explorer

Settings Features Replicate data globally Default consistency Firewall and virtual networks Private Endpoint Connections CORS Keys Add Azure Cognitive Search

STRONG BOUNDED STALENESS SESSION CONSISTENT PREFIX EVENTUAL

i Bounded staleness consistency is most frequently chosen by globally distributed applications expecting low write latencies but total global order guarantees.

Maximum Lag (Time)

Days	0	Hours	0
Minutes	0	Seconds	5

Maximum Lag (Operations)

100

Understand Bounded Staleness consistency

Unlike strong consistency which is scoped to a single region, you can choose bounded staleness consistency with any number of read regions (along with a write region). Bounded staleness is great for applications featuring group collaboration and sharing, stock ticker, publish-subscribe/queueing etc.

Setting the Consistency Level

Set default for entire account
Can be changed at any time

Override at the request level
Any request can weaken the default consistency level



Building Event-driven Microservices using the Change Feed

Agenda

Part I

Introduction to Azure Cosmos DB

- Overview
- Management
- Indexing
- Multiple APIs
- Migration
- Throughput
- Partitioning
- Global distribution

Part II

Building Event-driven Microservices with the Cosmos DB Change Feed

- Replication
- Denormalization
- Notifications
- Materialized views
- Data movement



Objectives

Cosmos DB

Review partitioning

Change feed

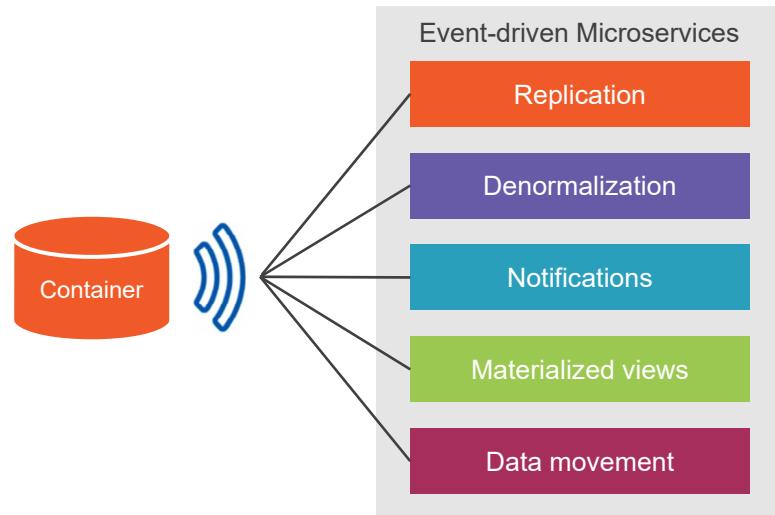
Learn about
Cosmos DB
change feed

Microservices

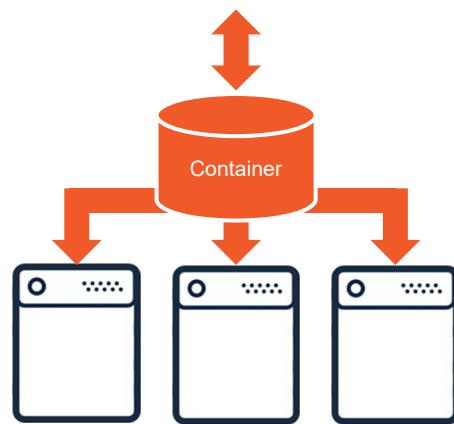
Apply these
concepts to build
event-driven
microservices



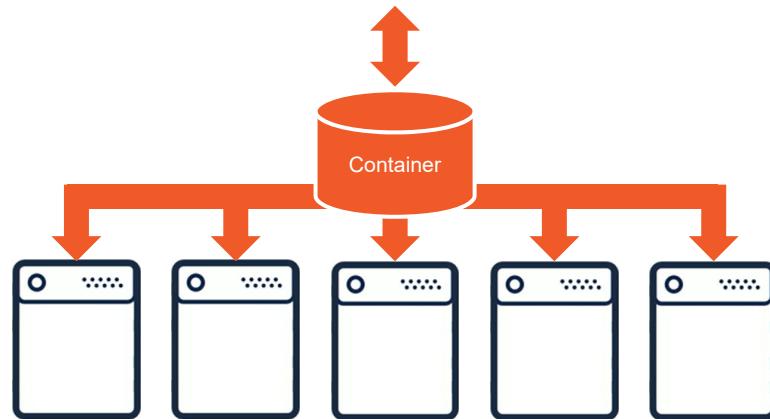
Introducing Change Feed



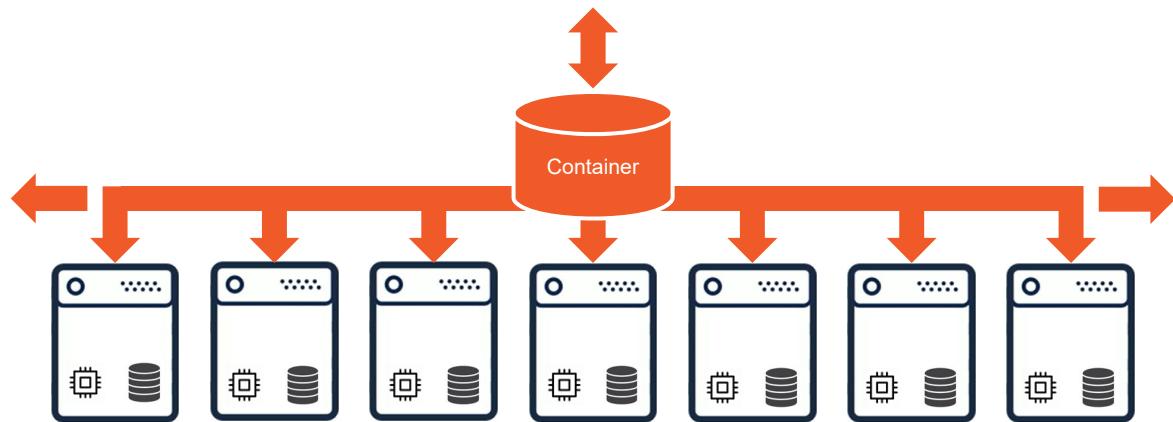
Horizontal Partitioning



Horizontal Partitioning



Horizontal Partitioning

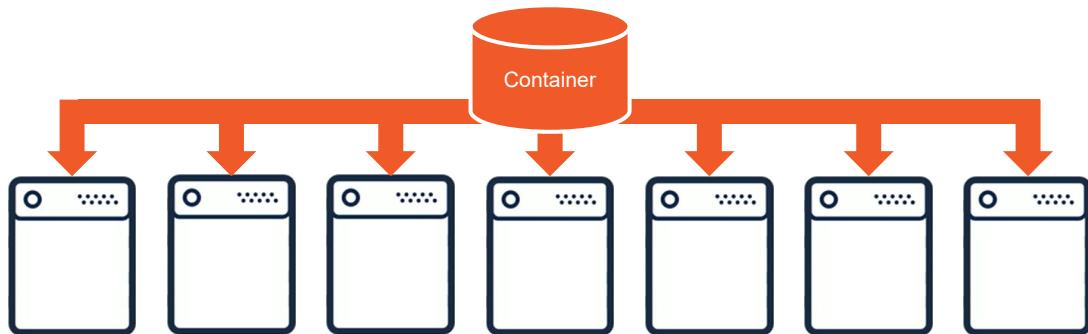


Unlimited storage

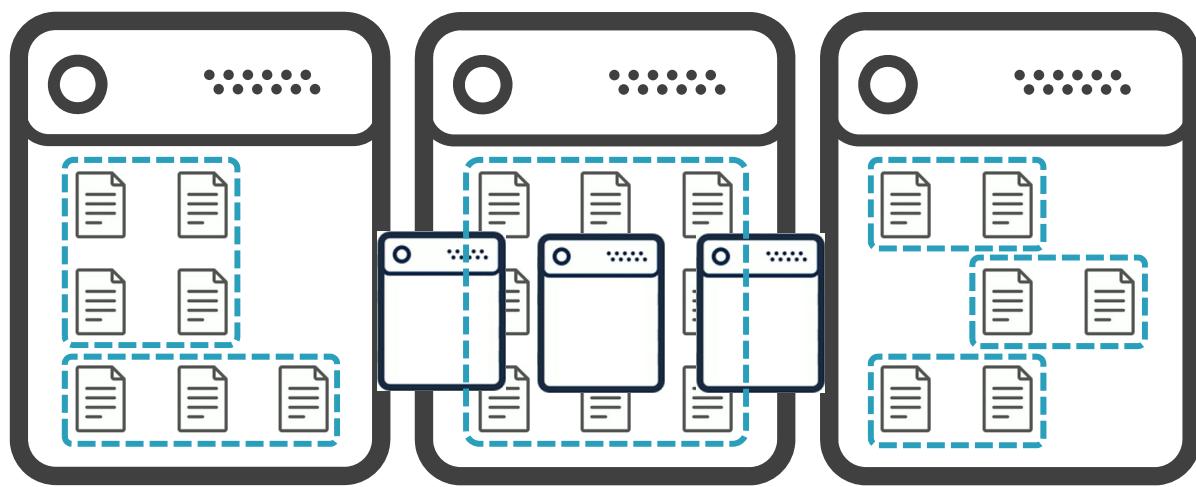
Unlimited throughput



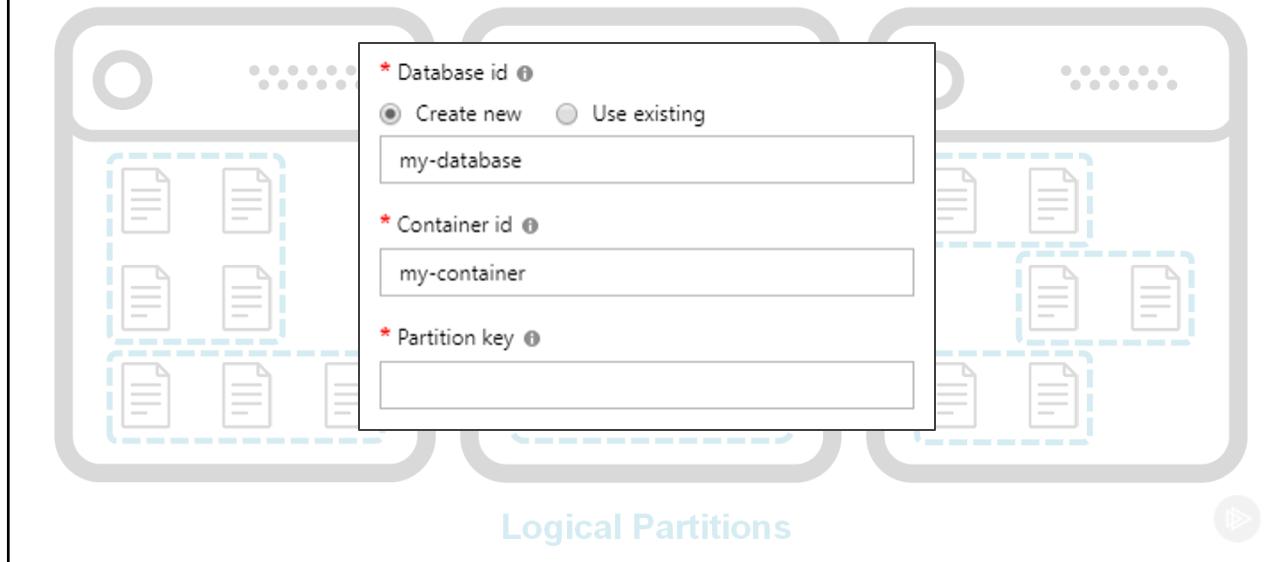
Understanding Logical Partitions



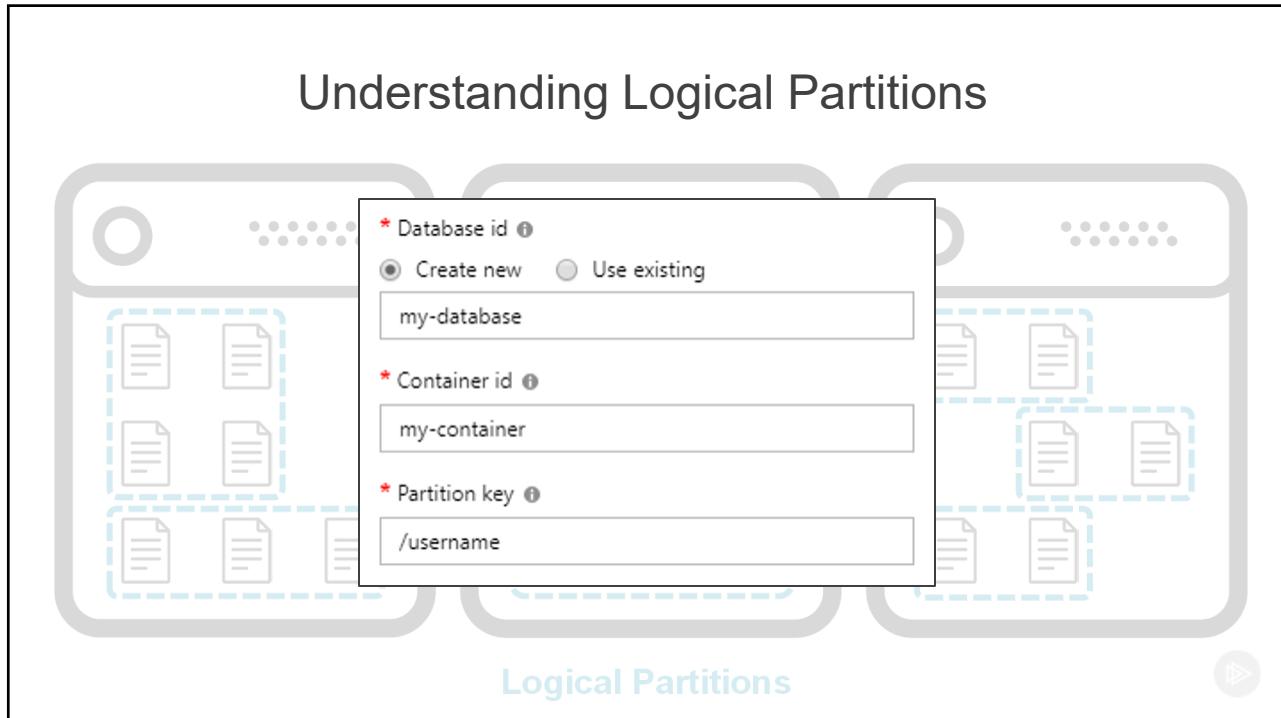
Understanding Logical Partitions



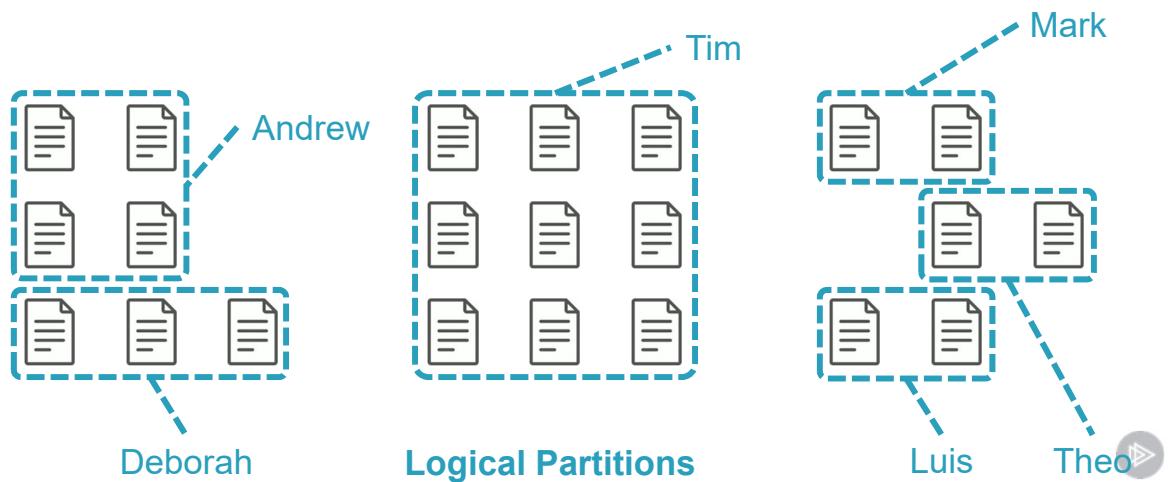
Understanding Logical Partitions



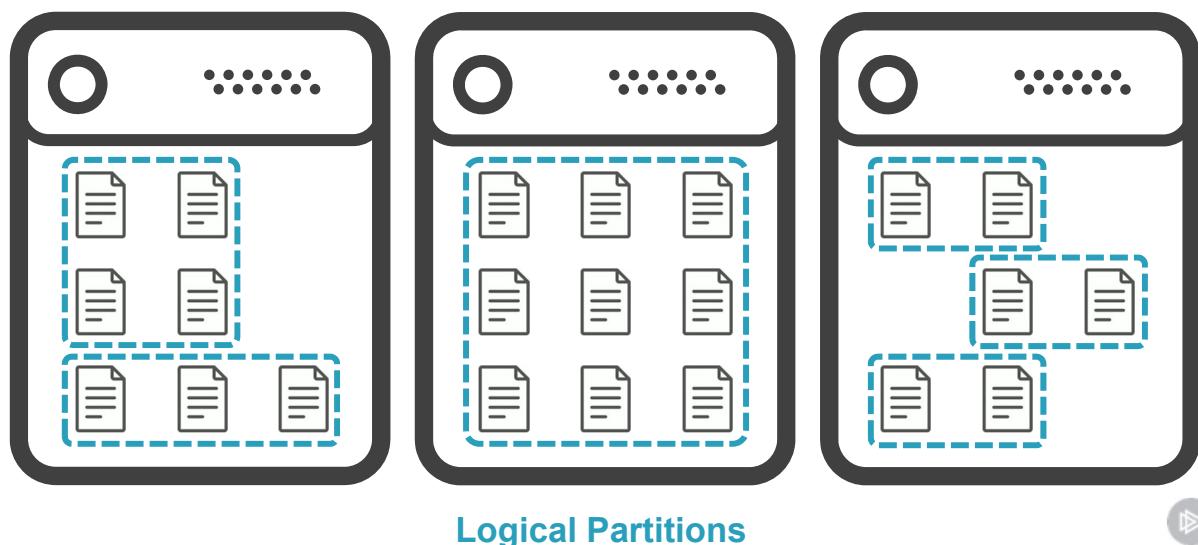
Understanding Logical Partitions



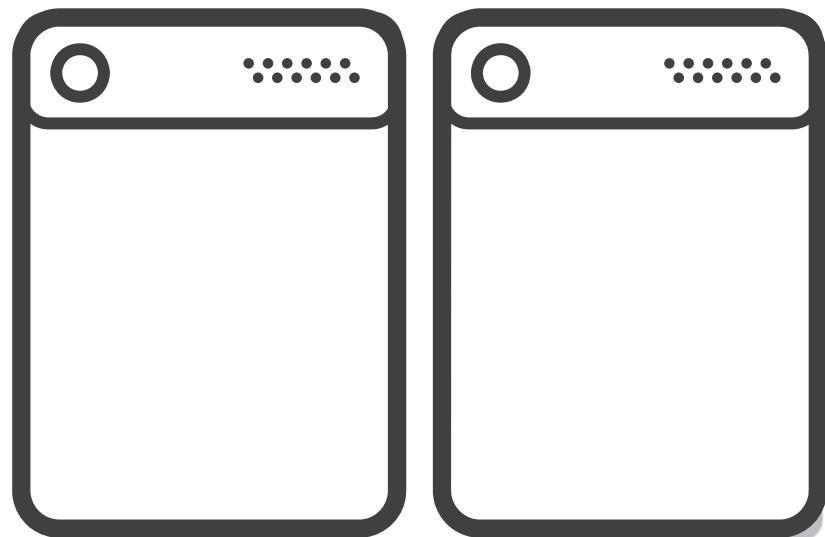
Understanding Logical Partitions



Understanding Logical Partitions

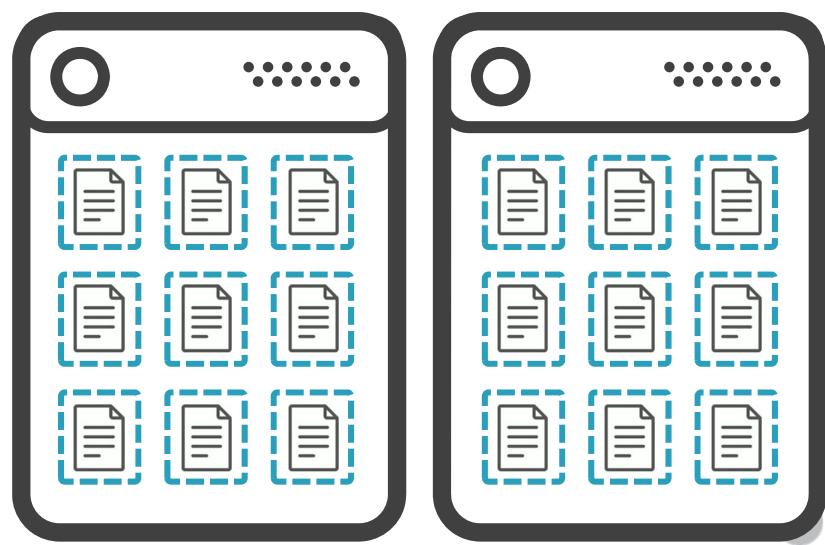


Common Partitioning Patterns



Common Partitioning Patterns

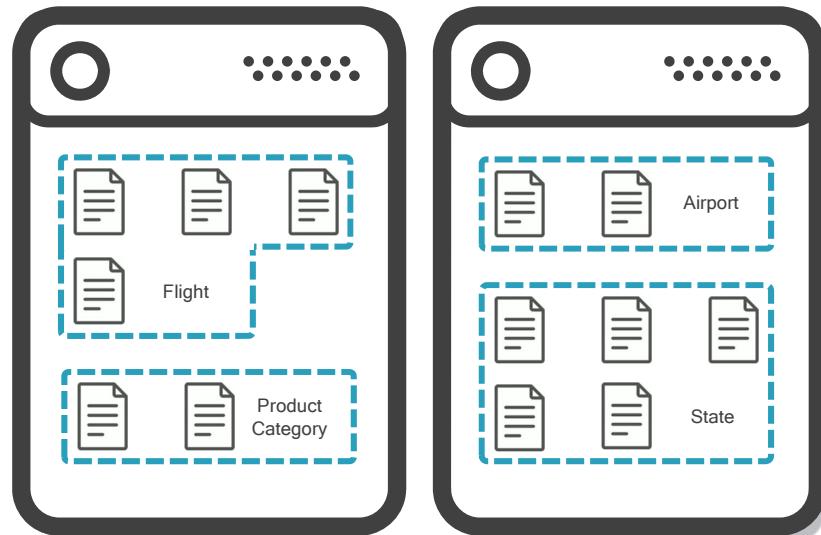
/id
Single-documents



Common Partitioning Patterns

/id
Single-documents

/type
Small lookup lists

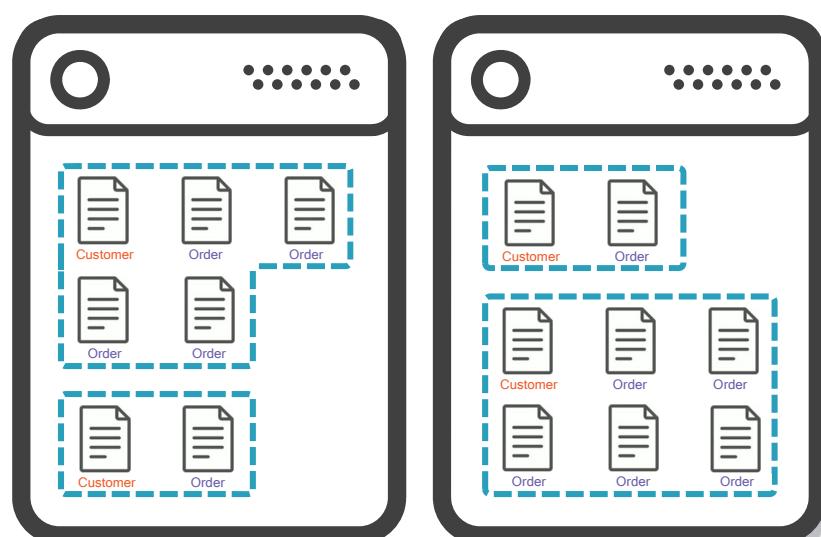


Common Partitioning Patterns

/id
Single-documents

/type
Small lookup lists

Other Example:
Optimize for queries /customerId

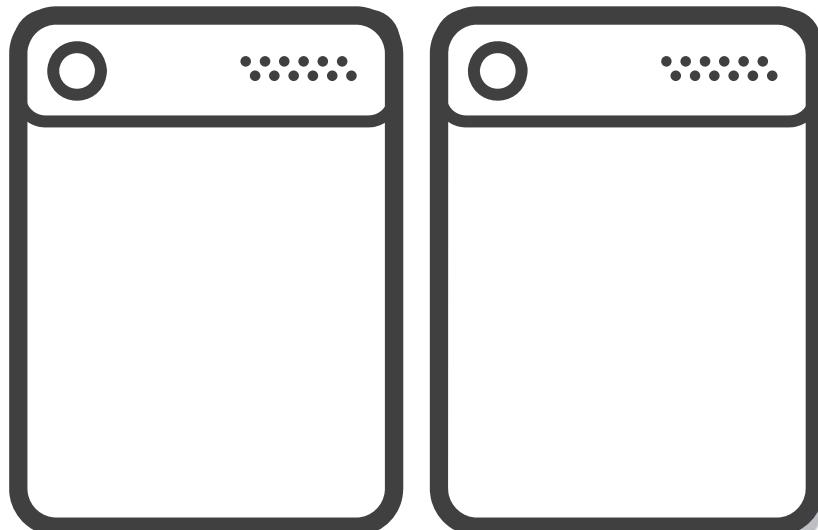


Common Partitioning Patterns

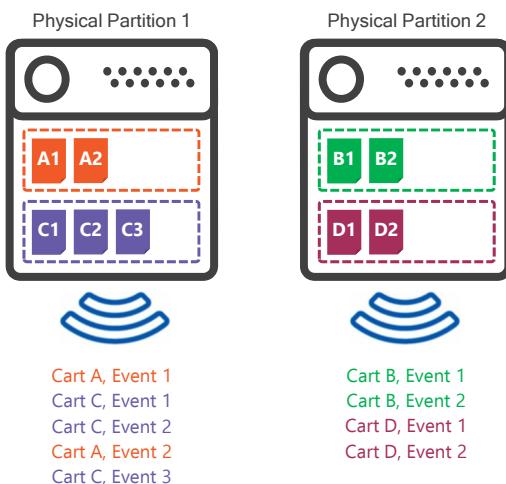
/id
Single-documents

/type
Small lookup lists

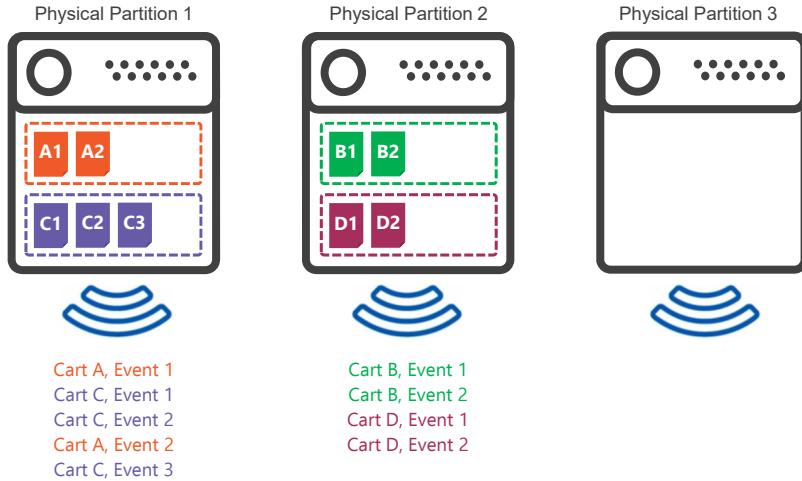
Other Example:
Optimize for queries /customerId



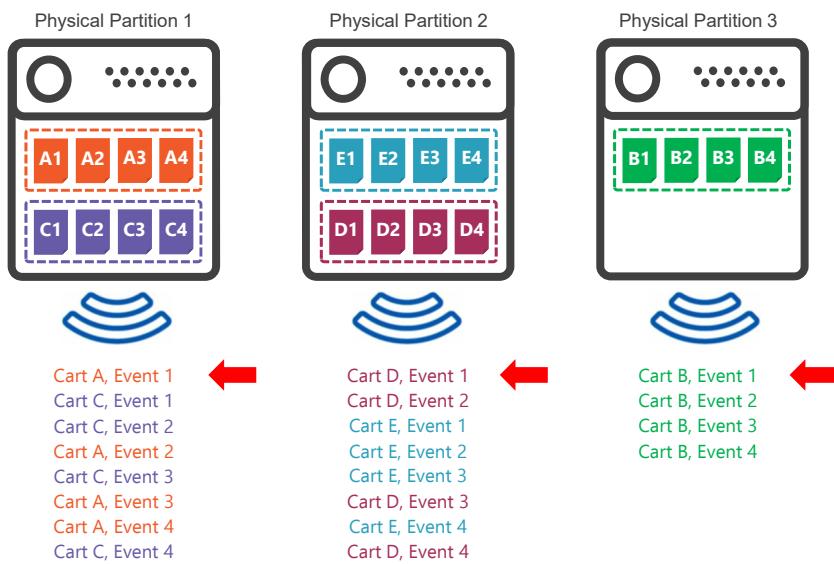
Ordered Change Events



Ordered Change Events



Ordered Change Events



Ordered Change Events

Physical Partition 1

Physical Partition 2

Physical Partition 3

Logical Change Feed

Cart A, Event 1
 Cart C, Event 1
 Cart C, Event 2
Cart A, Event 2
 Cart C, Event 3
Cart A, Event 3
Cart A, Event 4
 Cart C, Event 4

Cart D, Event 1
 Cart D, Event 2
 Cart E, Event 1
Cart A, Event 2
 Cart E, Event 2
 Cart E, Event 3
Cart D, Event 3
 Cart E, Event 4

Cart B, Event 1
 Cart B, Event 2
 Cart B, Event 3
Cart B, Event 4
 Cart B, Event 5

Cart A, Event 1
 Cart D, Event 1
 Cart C, Event 1
 Cart D, Event 2
Cart B, Event 1
 Cart D, Event 3
 Cart E, Event 1
 Cart C, Event 2
Cart A, Event 2
Cart B, Event 2
 Cart C, Event 3
 Cart E, Event 2
Cart A, Event 3
 Cart E, Event 3
Cart B, Event 3
Cart A, Event 4
 Cart E, Event 4
Cart B, Event 4
Cart D, Event 4
 Cart C, Event 4



Consuming the Change Feed

Directly
 Low-level direct
 access, per
 partition

Change Feed
 Processor (CFP)
 Library
 Stateful and
 scalable

Azure Functions
 Serverless wrapper
 around the CFP
 Library



Curing Analysis Paralysis

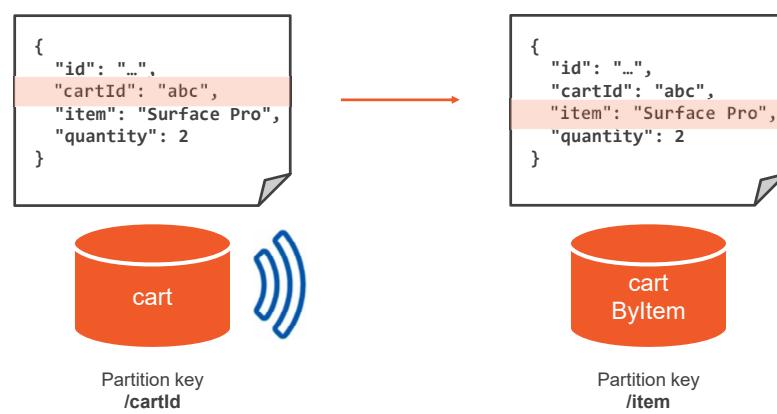
Choose a partition key
Based on your most common querying patterns

Only ONE partition key
What if you have more than just one common querying pattern?

Replication
Synchronize multiple containers with the same data, partitioned differently



Using Change Feed for Replication



```
SELECT * FROM c WHERE c.cartId = 'abc'
```

```
SELECT * FROM c WHERE Partition key = 'Surface Pro'
```



Processing Updates and Deletes

Updates

Interim updates are not retained by the change feed

Deletes

Deletes are not written to the change feed



Processing Updates and Deletes

```
{  
  "id": "...",  
  "cartId": "abc",  
  "item": "Surface Pro",  
  "quantity": 2  
}
```



Partition key
/cartId



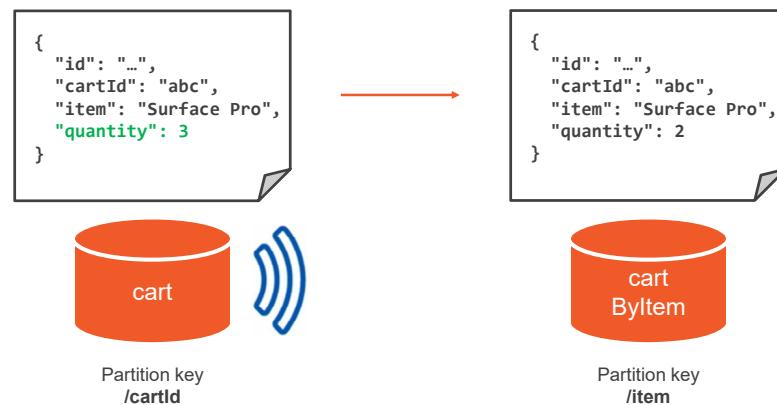
```
{  
  "id": "...",  
  "cartId": "abc",  
  "item": "Surface Pro",  
  "quantity": 2  
}
```



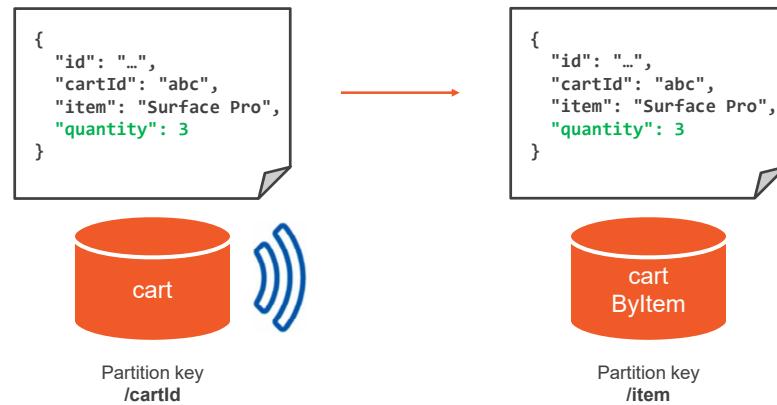
Partition key
/item



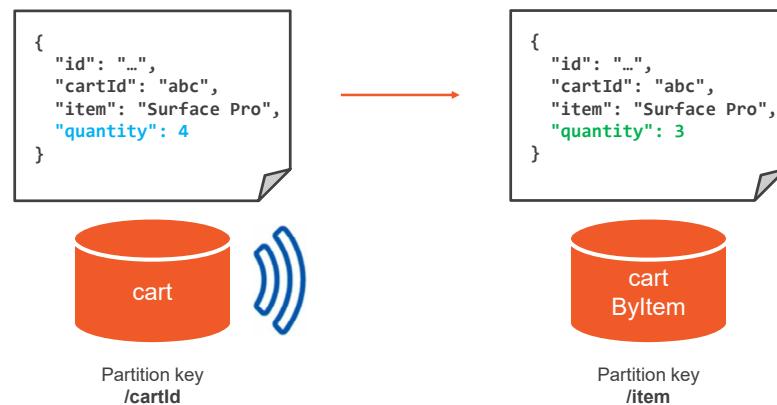
Processing Updates and Deletes



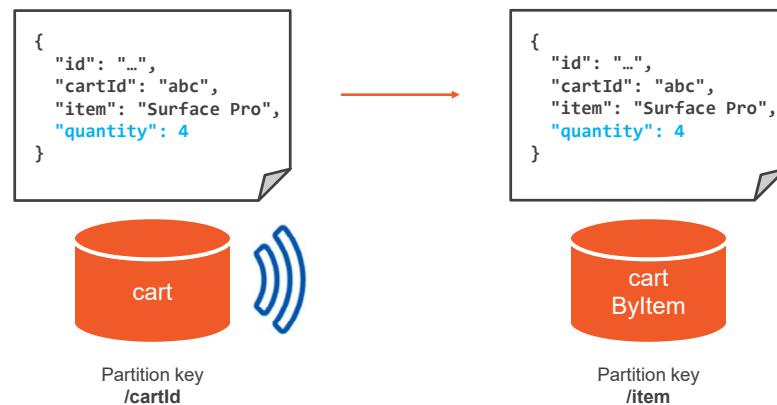
Processing Updates and Deletes



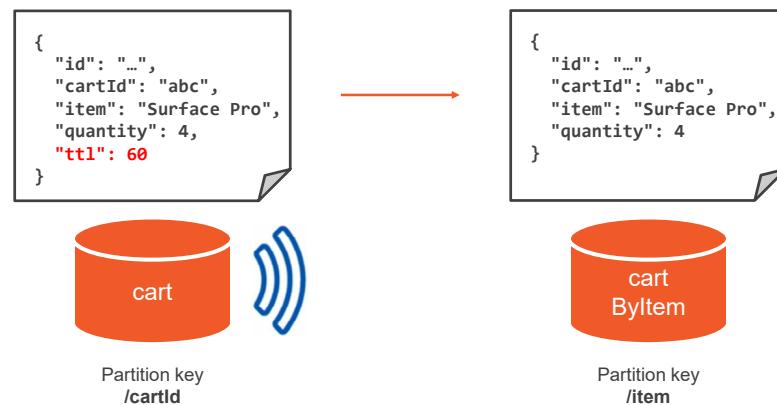
Processing Updates and Deletes



Processing Updates and Deletes



Processing Updates and Deletes



Full Fidelity Change Feed

Currently in Private Preview

Signup:
<https://forms.office.com/r/zFYSrQuJVV>

	Item Log	Operation Log
Operations	Inserts / latest updates only	All inserts, updates, and deletes
Retention	Infinite	Finite (max 30 days)
Optional metadata	None	Operation type, previous item version, delete via TTL



Using Azure Functions

CFP Library

Host is deployed to an Azure service

E.g., web job, worker role

Dedicated capacity

Azure Functions

Serverless, reactive functions

Shared capacity

Dedicated capacity with AF Premium



Demo



Replication Microservice



Handling Relational Workloads

Cosmos DB is non-relational

No JOINs
No relational constraints

Paradigm shift

Need to use different techniques to implement relations between entities

Denormalize

Duplicate specific properties
Satisfy queries with a single request



Querying Products

```
{
  "id": "...",
  "categoryId": "...",
  "sku": "...",
  "name": "...",
  "price": "...",
  "tagIds": [
    ...
  ]
}
```

Category name?

Tag names?



SELECT * FROM c WHERE c.categoryId = '<categoryId>'

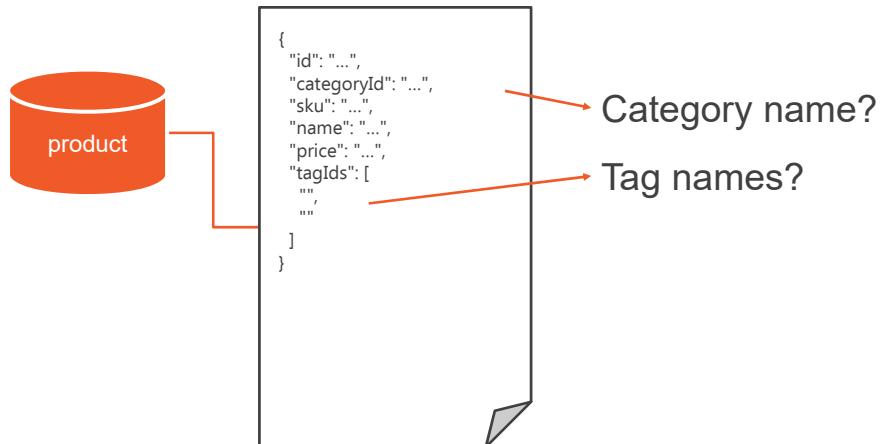


SELECT c.name FROM c WHERE c.type = 'category'
AND c.id = '<categoryId>'

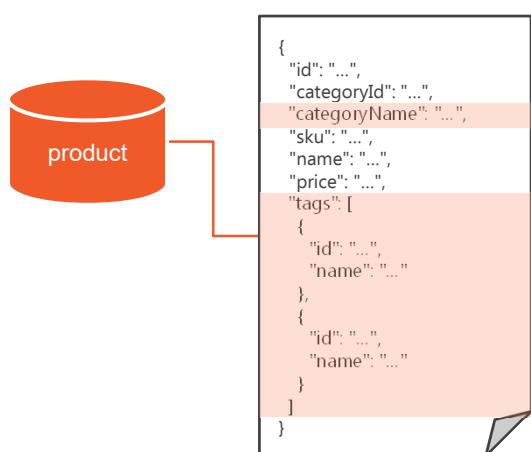
SELECT c.name FROM c WHERE c.type = 'tag'
AND c.id IN '<>tagId1>, <tagId2>, ..., <tagIdN>'



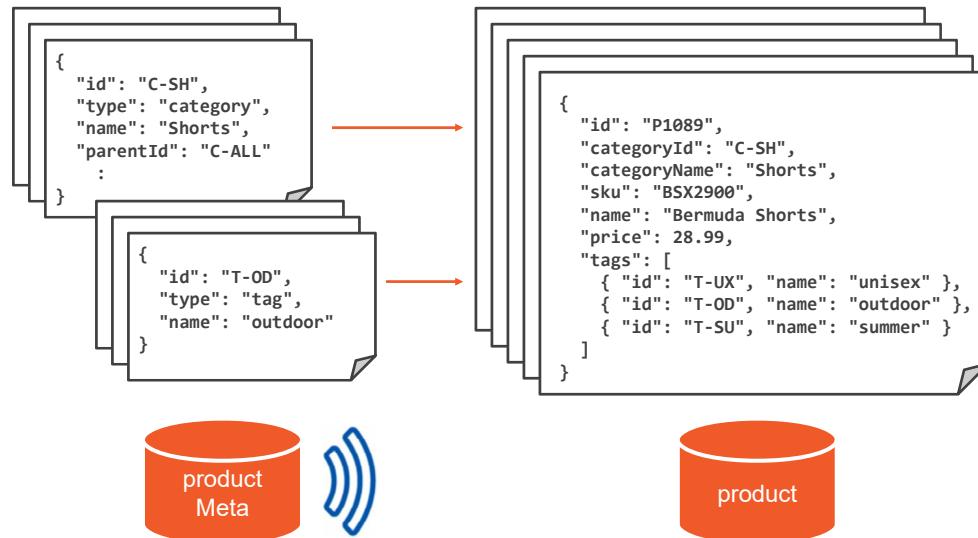
Denormalizing Products



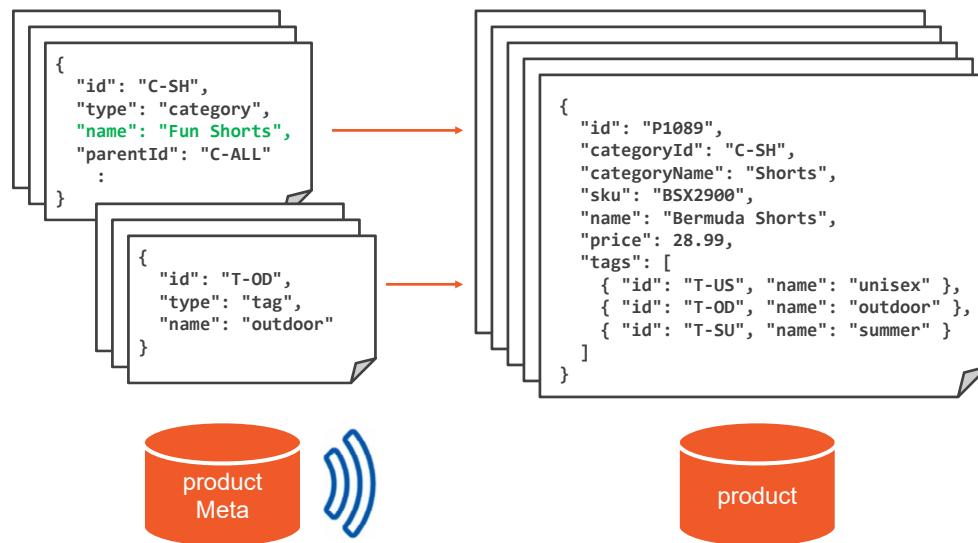
Denormalizing Products



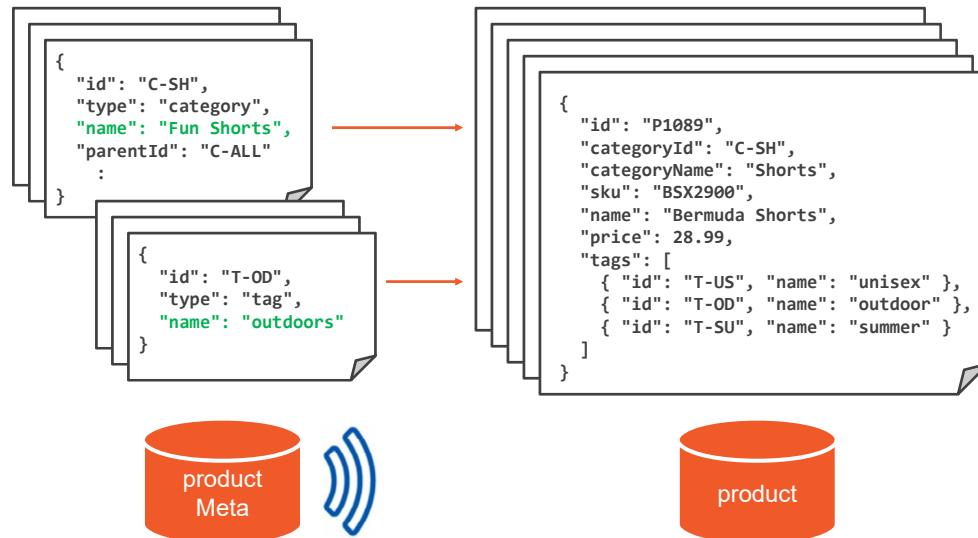
Denormalizing Products



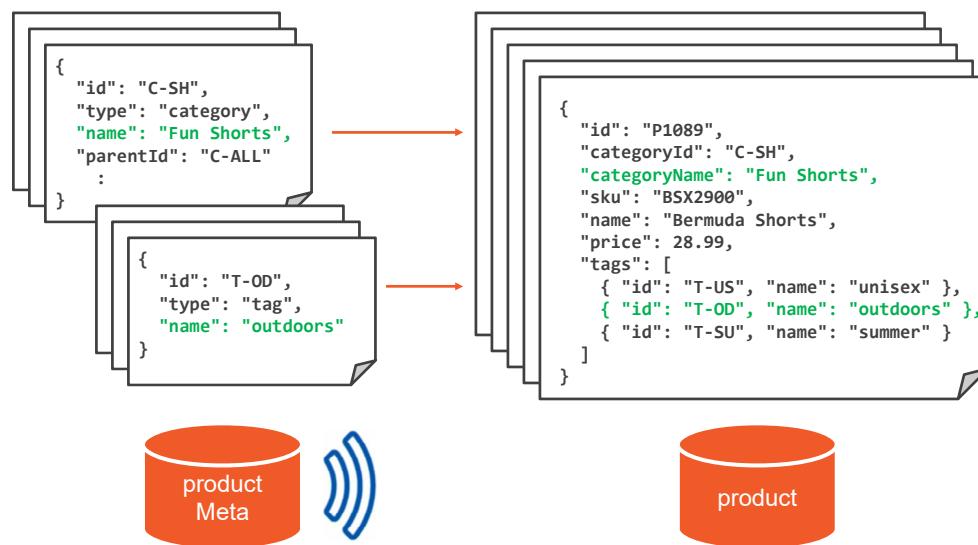
Denormalizing Products



Denormalizing Products



Denormalizing Products



Demo



Denormalization Microservice



Contoso Airlines IoT Scenario

We're a
small airline

Continental U.S.

12 flights

6 airports

Requirements

Real-time flight info

No-fly zone alerts

Up-to-date arrivals

Permanent archival

Solution

Ingest flight telemetry

Implement microservices

Email alerts

Materialized views

Archive to Blob Storage



Ingesting Flight Telemetry

location container

Receives flight telemetry

Speed, altitude, duration,
location (lat/long)

Every 10ms, per flight

Partitioned on /id

Exactly one document per
logical partition

Optimized for bulk loading
of device telemetry in real-
time



Demo



Flight Telemetry Generator



No-fly Zone Alerts

Track all flights

Constantly monitor each flight location

Issue alerts when a flight enters a no-fly zone

Monitor the change feed

Examine each telemetry document

Run a spatial query against designated no-fly zones



Demo



Email Alert Microservice



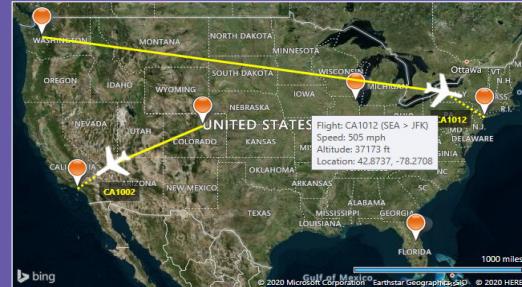
Querying Real-time Flight Data

Get current telemetry

Query the location container for each flight in the air

```
SELECT TOP 1 *
FROM c
WHERE
c.flightNumber = 'CA1012'
ORDER BY
c._ts DESC
```

Present a live map
Bing Maps mashup



Demo



Flight Location Map



Introducing Materialized Views

Get current location

Continuous querying for each flight's current telemetry

Expensive operation over voluminous raw data using cross-partition query

Build a materialized view

Materialize a tiny view – one document per flight

Partition on /type

Set /id to flight number

Extremely cheap to query the materialized view



Demo



Current Location Microservice



Querying Arrival Data

Get current telemetry
 Query the remaining time for each flight in the air
 Group the flights by airport
 Create a materialized view

```
{
  "id": "LAX",
  "type": "arrival",
  "arr": [
    {
      "fli": [
        {
          "id": "SEA",
          "type": "arrival",
          "arr": [
            {
              "fli": [
                {
                  "id": "MCO",
                  "type": "arrival",
                  "arrivalAirport": "MCO",
                  "flights": [
                    {
                      "flightNumber": "CA1008",
                      "departureAirport": "ORD",
                      "remainingMinutes": 88.8
                    },
                    {
                      "flightNumber": "CA1007",
                      "departureAirport": "LAX",
                      "remainingMinutes": 235.8
                    },
                    {
                      "flightNumber": "CA1007",
                      "departureAirport": "LAX",
                      "remainingMinutes": 235.8
                    }
                  ]
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
```



Querying Arrival Data

Get current telemetry
 Query the remaining time for each flight in the air
 Group the flights by airport
 Create a materialized view

Present an arrivals board

Query the materialized view

ARRIVALS			
Flight	From	To	Status
CA1001	JFK	LAX	10:49 PM
CA1002	DEN	LAX	ARRIVED
CA1003	ORD	LAX	09:33 PM
CA1005	MCO	LAX	10:06 PM
CA1010	SEA	LAX	ARRIVED



Demo



Arrivals Board Microservice



Data Movement

Replicate to Secondary
Data Store

Feed data changes to
downstream systems

SQL Database

Archive to Cold Storage

Use TTL to establish a
retention period

E.g., Azure Blob Storage,
Azure Data Lake



Demo



Data Archival Microservice



Using the Pull Model Alternative

Consume at
your own pace

Get all changes,
or by partition key

Specify an
optional start time

When to use
the pull model?

Reading changes by
partition key

Controlling the pace

One-time migrations



Thank You!

Contact Me

lenni.lobel@sleektech.com

Visit my blog

lennilobel.wordpress.com

Follow me on Twitter

[@lennilobel](https://twitter.com/lennilobel)

Thanks for coming! ☺

