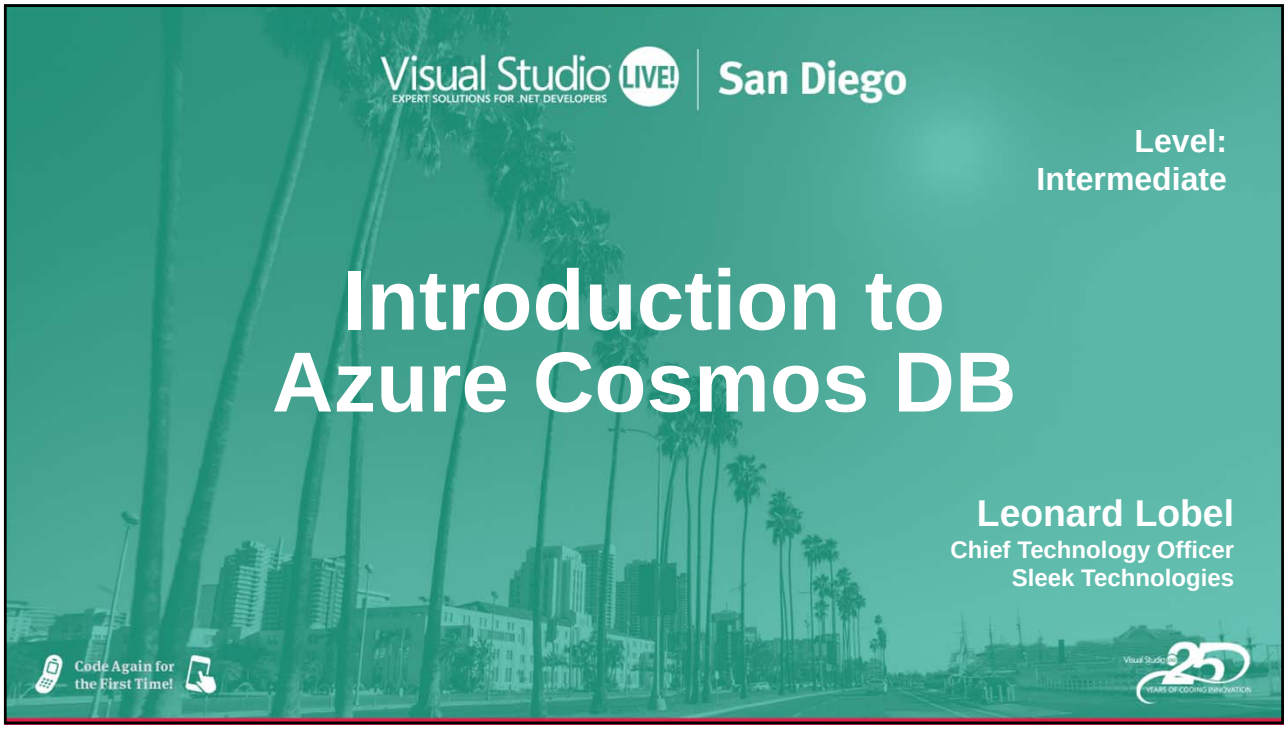


About Me

sleek technologies

TALLAN

MVP Microsoft Most Valuable Professional

pluralsight

Leonard Lobel

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

Contact

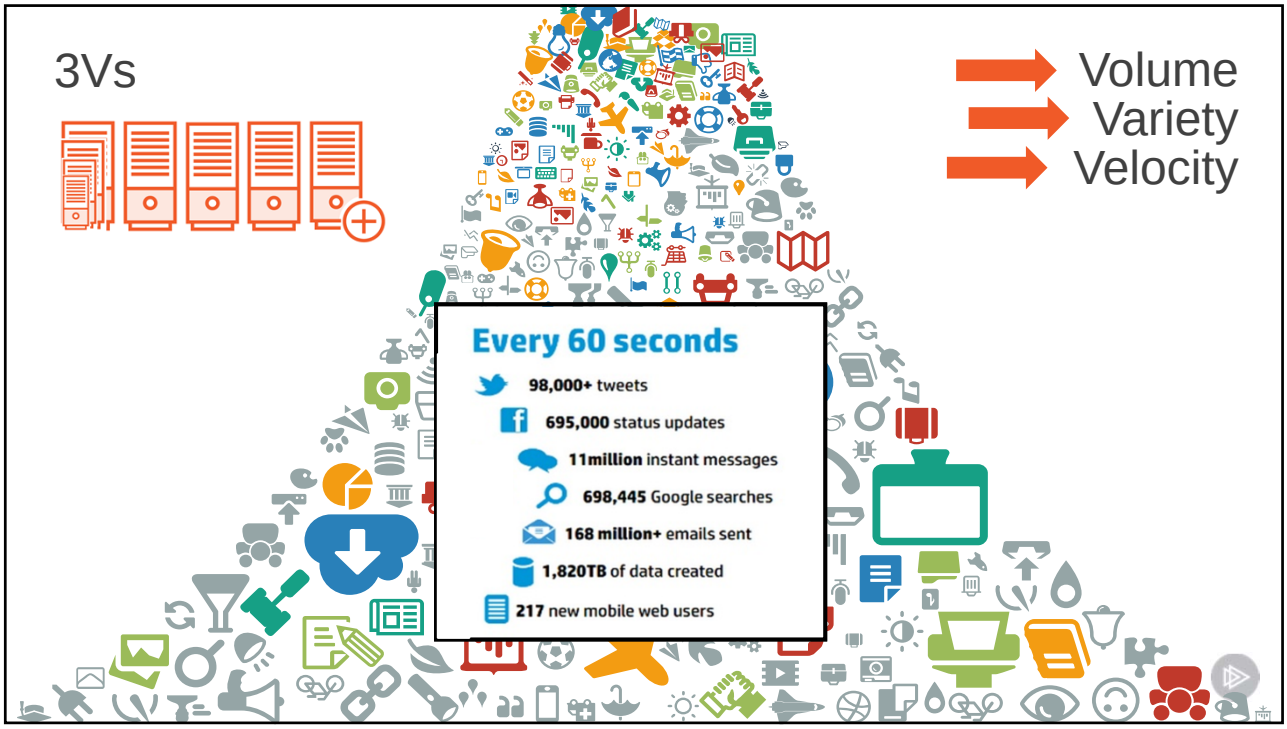
- Email: lenni.lobel@sleektech.com
- Blog: lennilobel.wordpress.com
- Twitter: @lennilobel

Download Slides

http://bit.ly/vslboston2018_cosmosdb

(all lower case!)

What is NoSQL?



What is a NoSQL database?

Distributed

Replicas ensure high throughput/availability, and low latency

Scale-out

Horizontal partitioning enables virtually limitless storage and throughput

Schema-free

Document, table, graph, and columnar data models

What is Cosmos DB?

Evolution of DocumentDB

Scalable NoSQL document database
Low latency (single-digit millisecond)

Virtually unlimited scale

Scale storage with server-side partitioning
Scale throughput with variable request units

Turnkey global distribution

Point-and-click control over where your data gets geo-replicated

Multi-model / Multi-API

No longer exclusively a document database
Also supports tables, graph, and columnar

Cosmos DB Evolution

2010 - 2014 — Internal Microsoft DocumentDB service
Office, OneNote, Xbox

Aug 21, 2014 — Azure DocumentDB preview
SQL grammar over schema-free JSON
Tunable throughput, indexing, consistency
Server-side ACID transactions
Runs on Azure (fully managed PaaS)

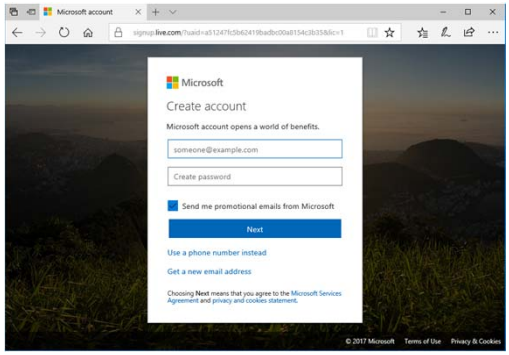
Apr 8, 2015 — Azure DocumentDB GA (General Availability)

Apr - Aug 2015 — New features
ORDER BY and string range queries
Geospatial support
SDK partitioning support

Mar 31, 2016 — New features
Partitioned collections
Geo-replication
Wire support for MongoDB (BSON)

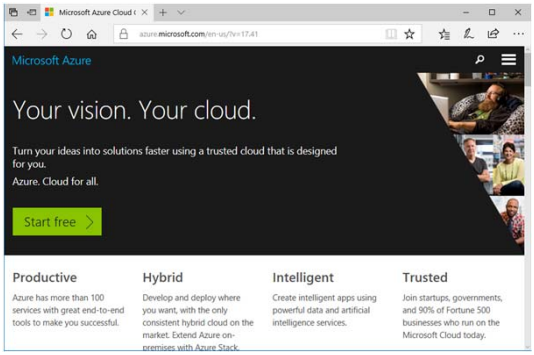
May 10, 2017 — Azure Cosmos DB
Turnkey global distribution
Horizontal partitioning
Low latency (single-digit millisecond)
SLAs for guarantees on 99.99% availability, throughput, latency, and consistency
Local emulator
Multi-model, multi-API database
- JSON (SQL API)
- BSON (MongoDB API)
- Graph (Gremlin API)
- Key-value (Table API)
- Columnar (Cassandra API)

Getting Started



Get a Microsoft account
http://signup.live.com

+



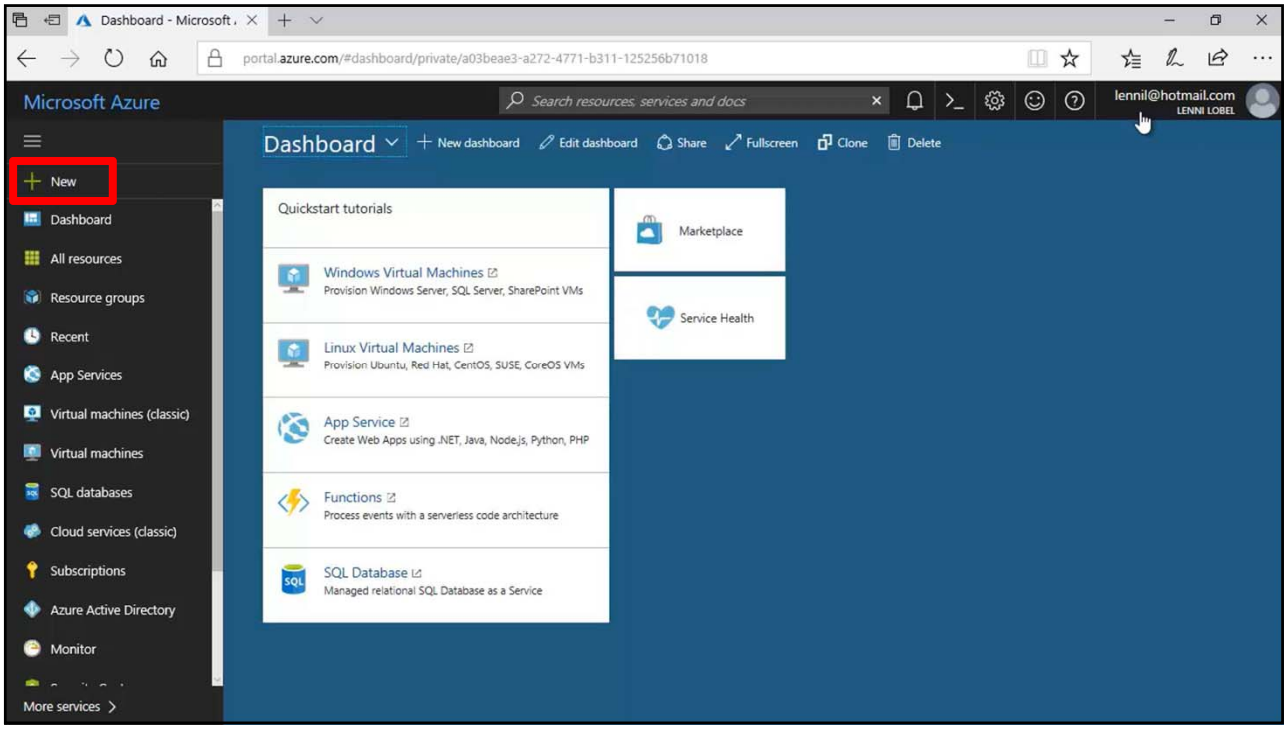
Get a Microsoft Azure subscription
http://azure.microsoft.com

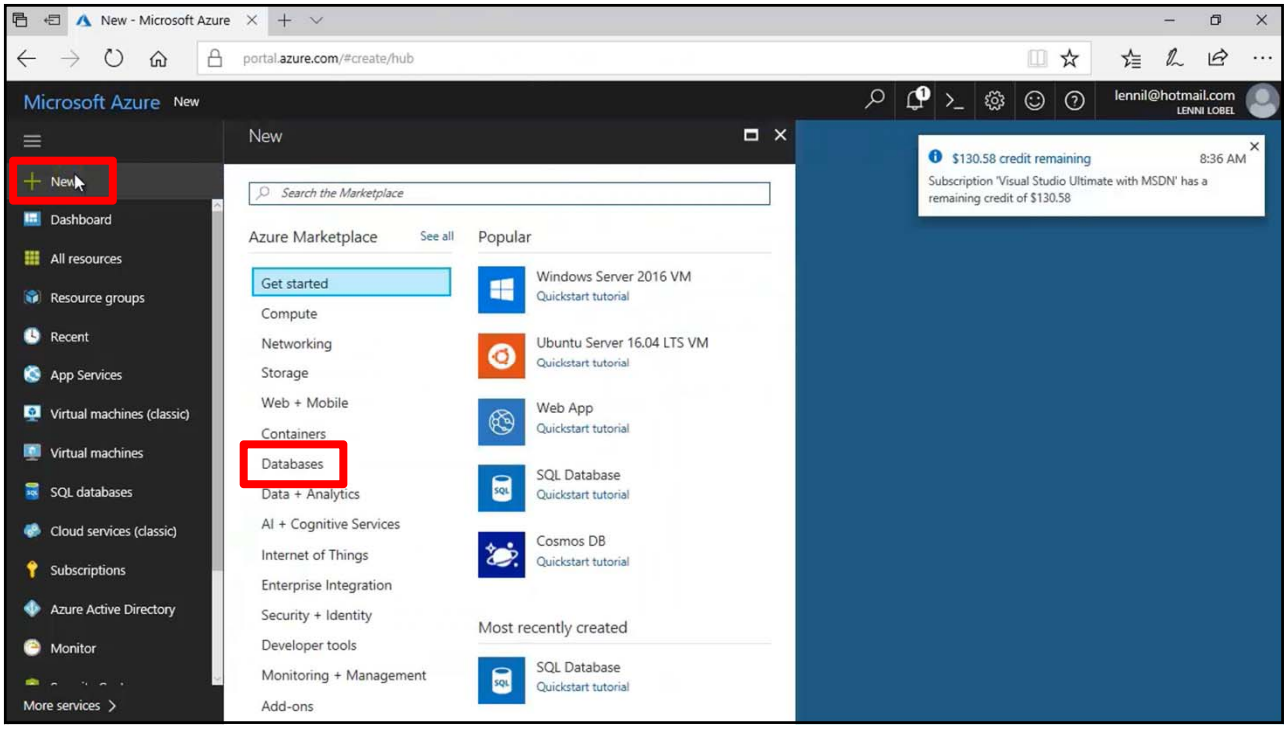
Or,

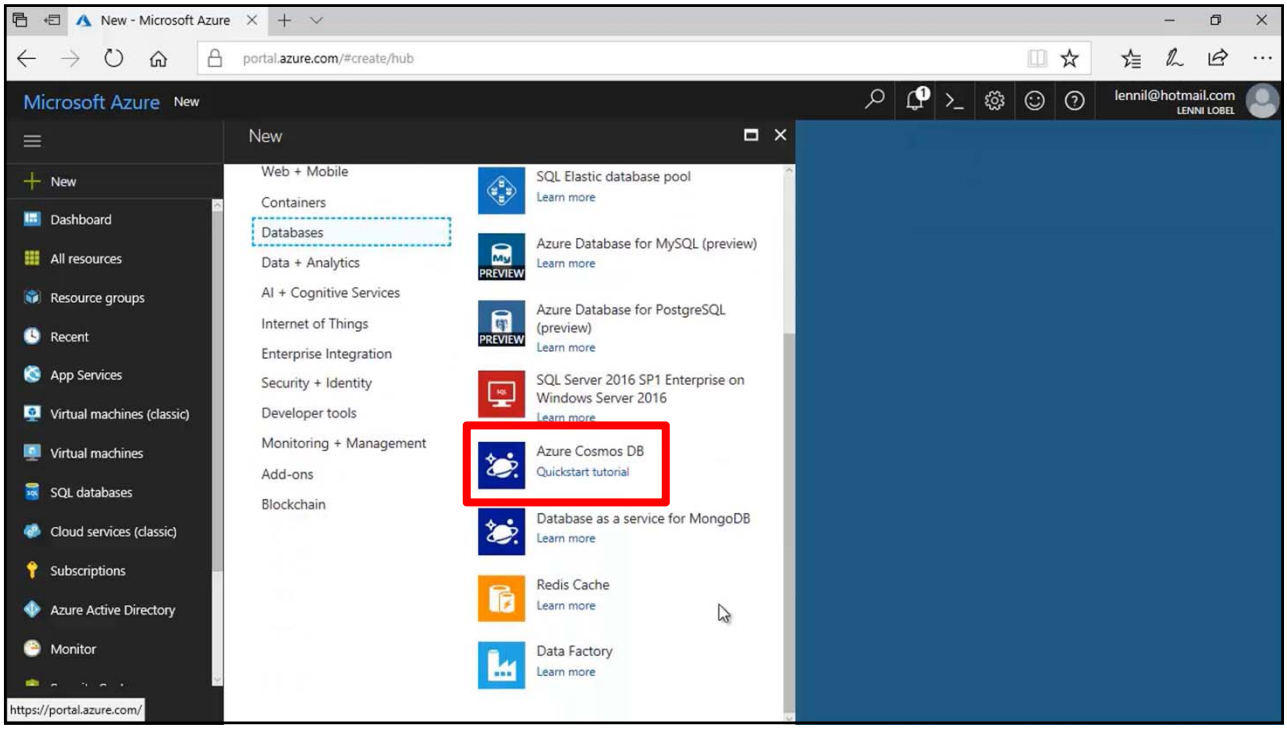
Download the local emulator
https://aka.ms/cosmosdb-emulator

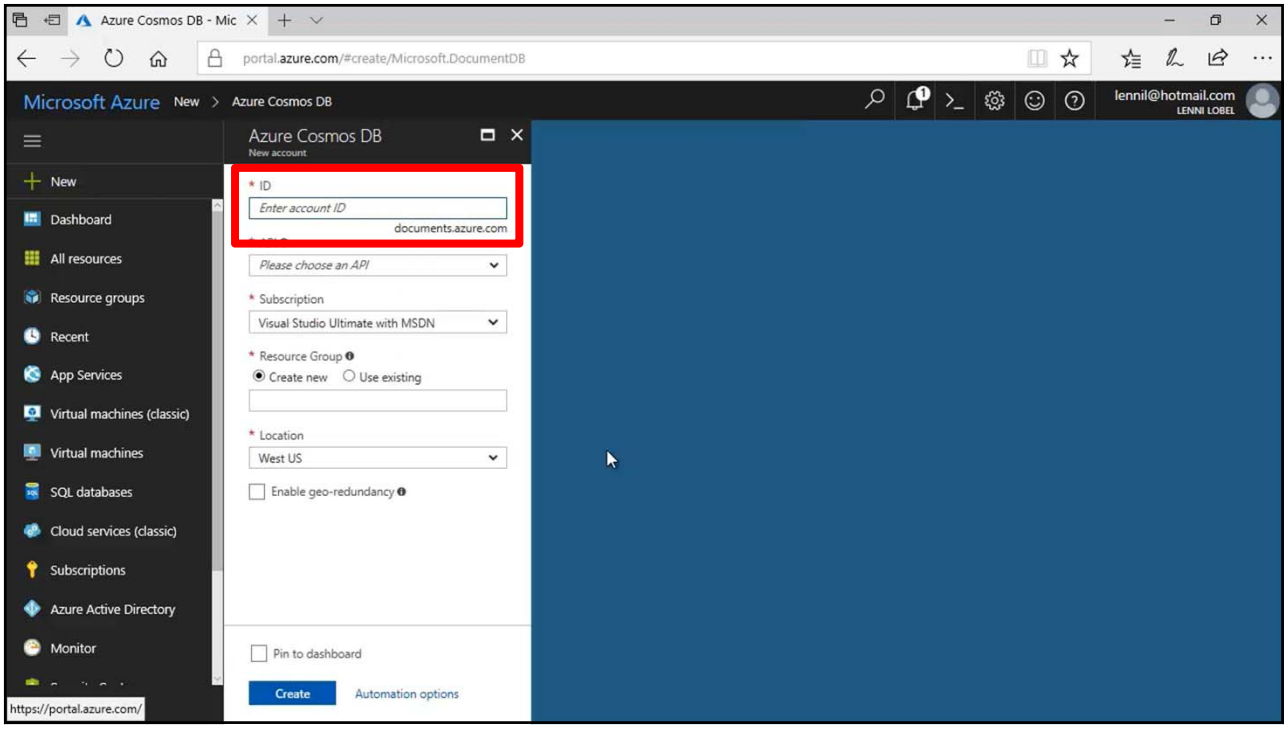
Demo

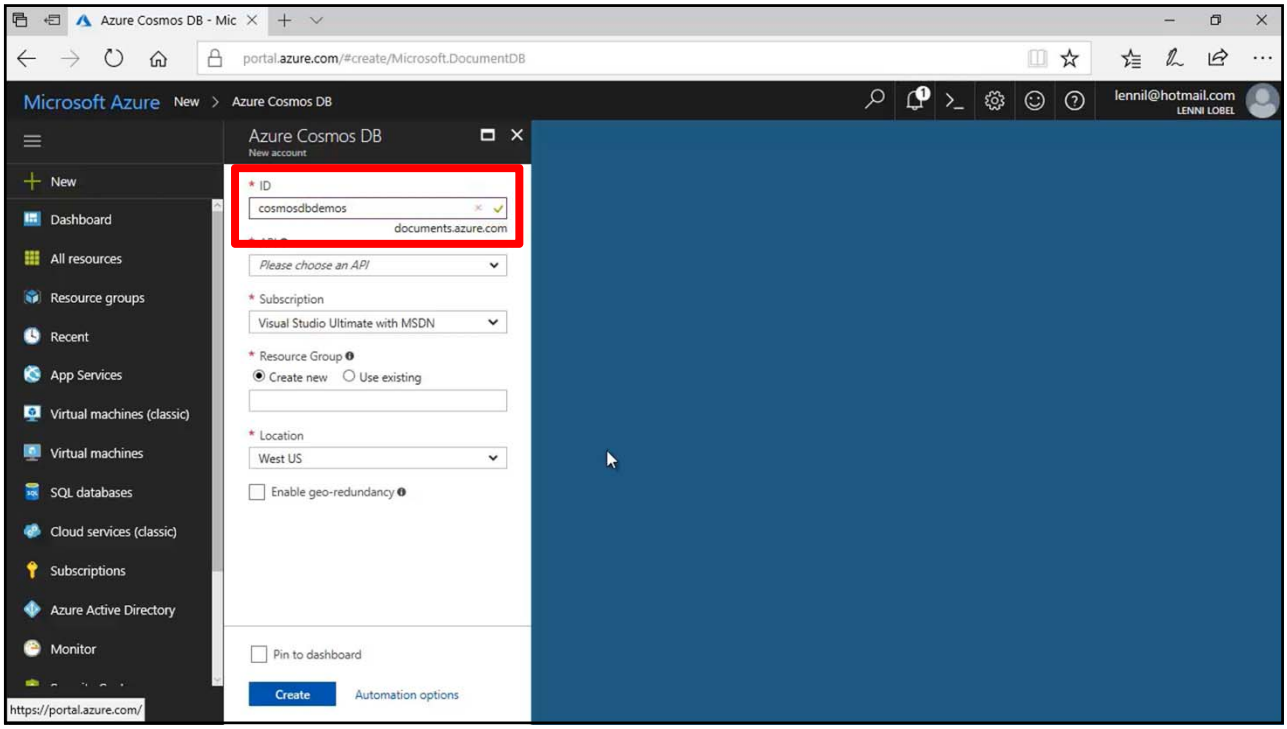
Creating a Cosmos DB account

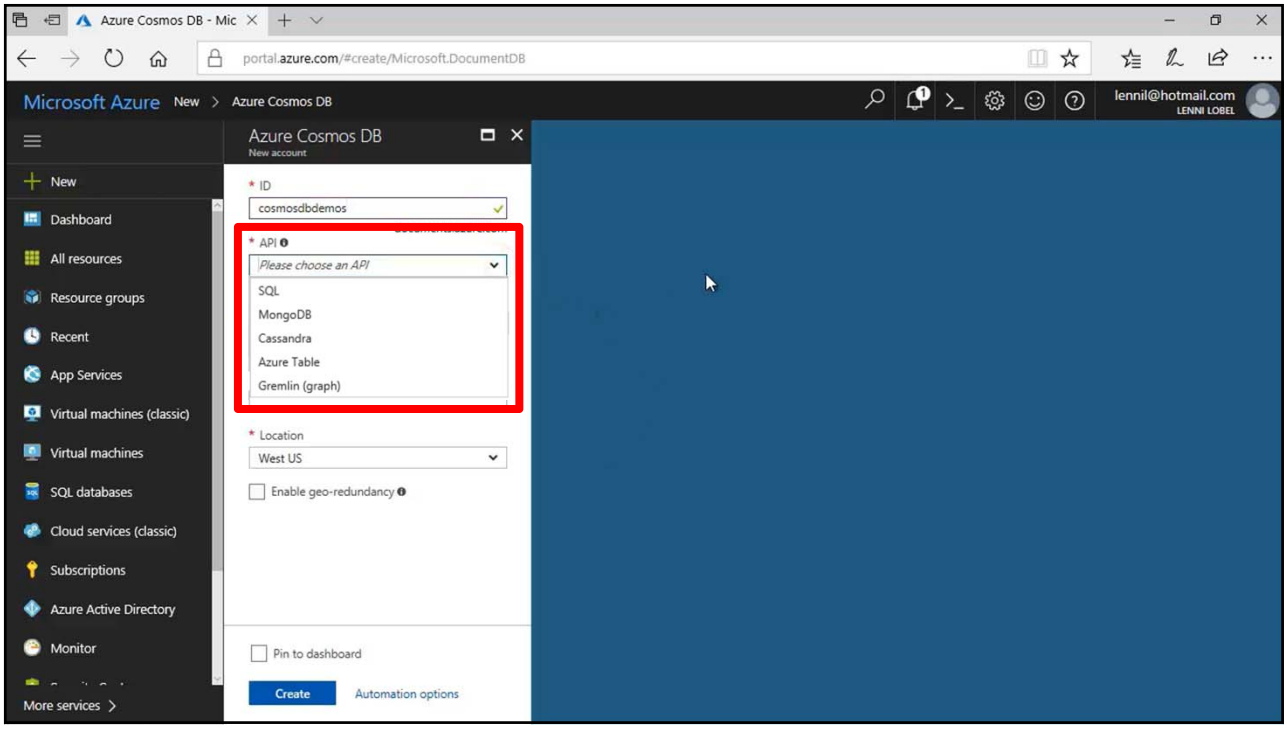

Dashboard - Microsoft
portal.azure.com/#dashboard/private/a03beae3-a272-4771-b311-125256b71018
Microsoft Azure
Search resources, services and docs
lennil@hotmail.com
LENNI LOBEL
Dashboard
New dashboard
Edit dashboard
Share
Fullscreen
Clone
Delete
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
More services
Quickstart 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
Marketplace
Service Health

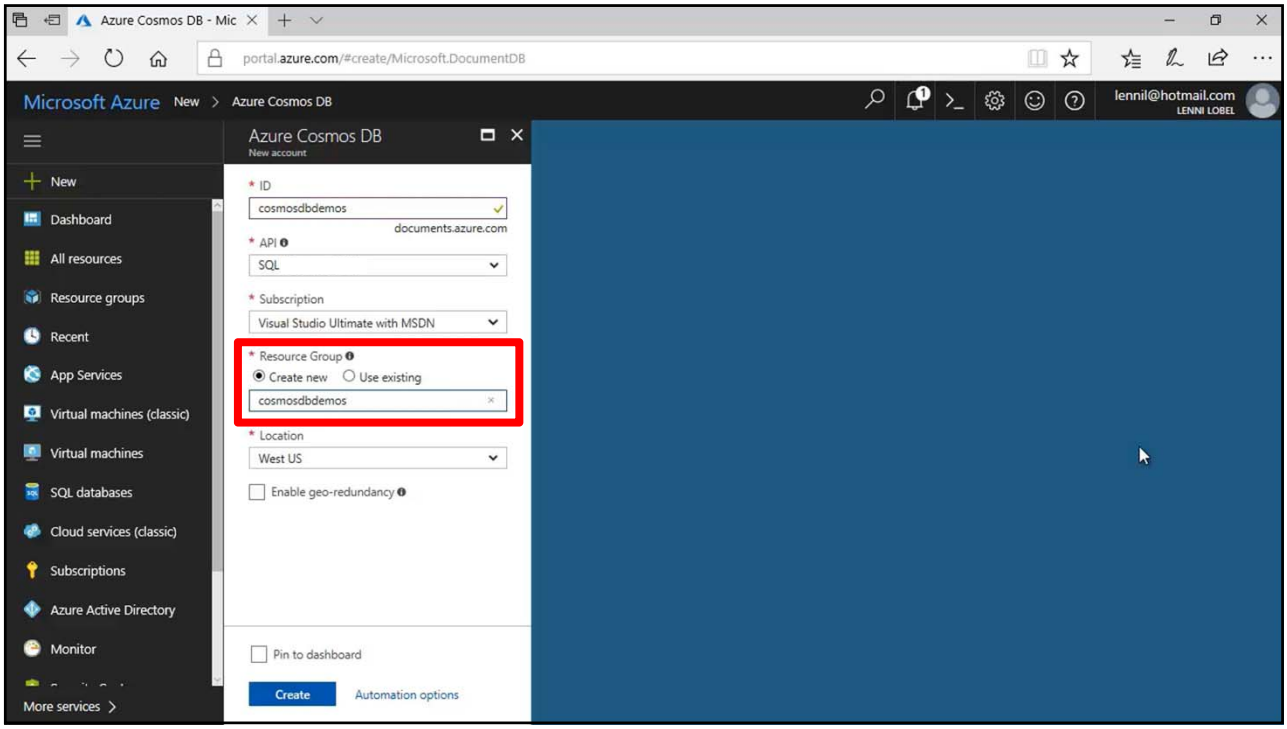

New - Microsoft Azure
portal.azure.com/#create/hub
Microsoft Azure
New
lennil@hotmail.com
LENNI LOBEL
New
Search the Marketplace
Azure Marketplace
See all
Get started
Compute
Networking
Storage
Web + Mobile
Containers
Databases
Data + Analytics
AI + Cognitive Services
Internet of Things
Enterprise Integration
Security + Identity
Developer tools
Monitoring + Management
Add-ons
Popular
Windows Server 2016 VM
Quickstart tutorial
Ubuntu Server 16.04 LTS VM
Quickstart tutorial
Web App
Quickstart tutorial
SQL Database
Quickstart tutorial
Cosmos DB
Quickstart tutorial
Most recently created
SQL Database
Quickstart tutorial
$130.58 credit remaining
8:36 AM
Subscription 'Visual Studio Ultimate with MSDN' has a remaining credit of $130.58

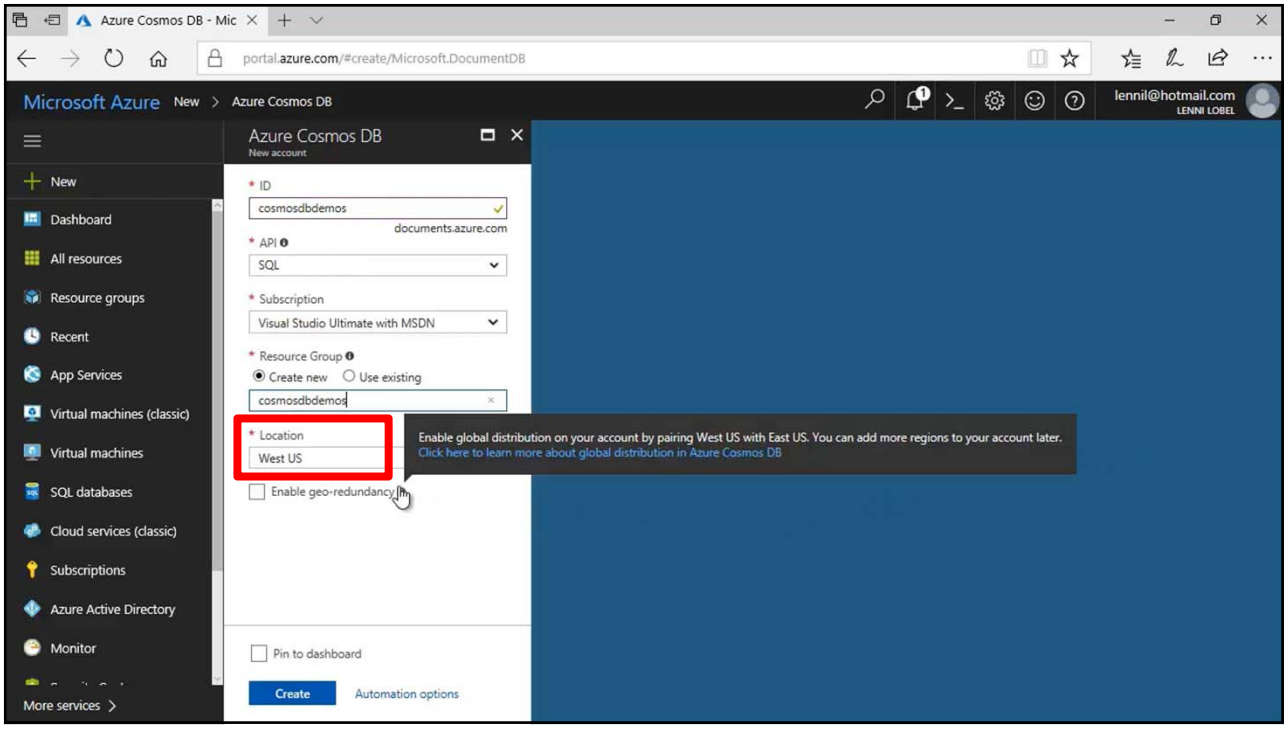

New - Microsoft Azure
portal.azure.com/#create/hub
Microsoft Azure
New
lennil@hotmail.com
LENNI LOBEL
New
Web + Mobile
Containers
Databases
Data + Analytics
AI + Cognitive Services
Internet of Things
Enterprise Integration
Security + Identity
Developer tools
Monitoring + Management
Add-ons
Blockchain
SQL Elastic database pool
Learn more
Azure Database for MySQL (preview)
Learn more
Azure Database for PostgreSQL (preview)
Learn more
SQL Server 2016 SP1 Enterprise on Windows Server 2016
Learn more
Azure Cosmos DB
Quickstart tutorial
Database as a service for MongoDB
Learn more
Redis Cache
Learn more
Data Factory
Learn more
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
https://portal.azure.com/

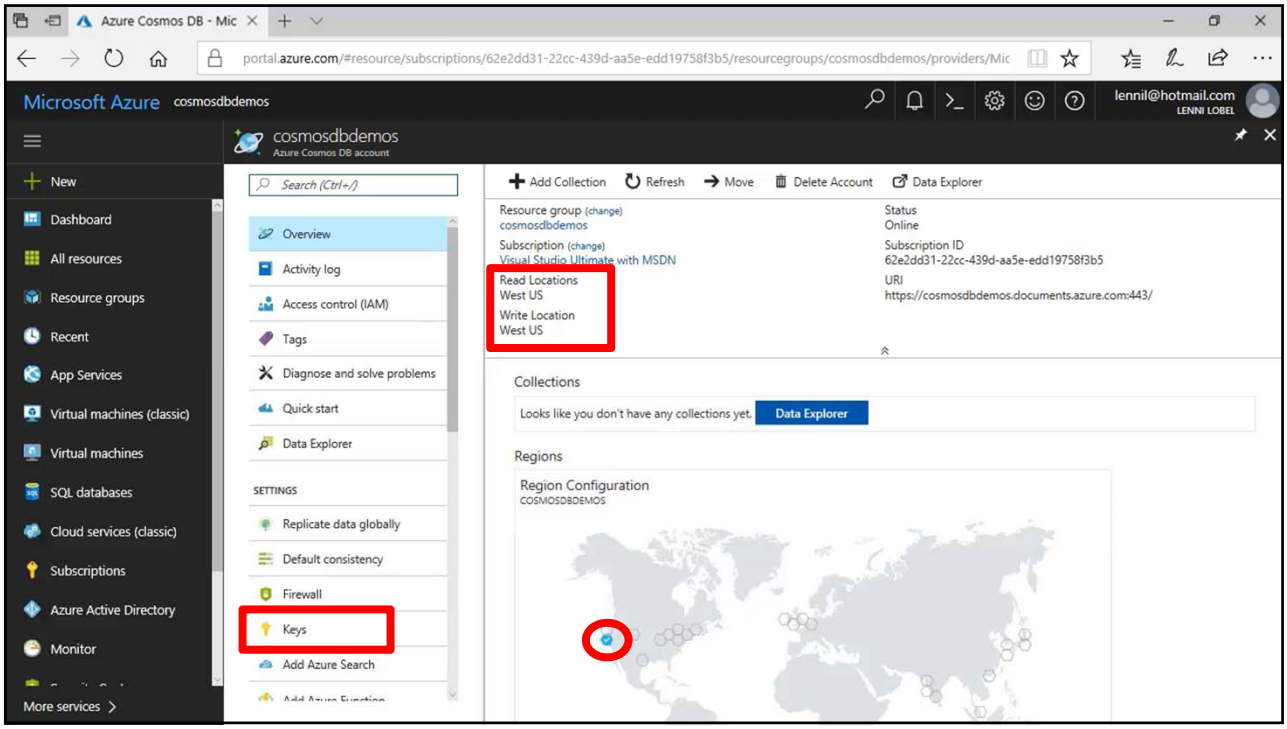

Azure Cosmos DB - Mic
portal.azure.com/#create/Microsoft.DocumentDB
Microsoft Azure
New > Azure Cosmos DB
lennil@hotmail.com
LENNI LOBEL
Azure Cosmos DB
New account
* ID
Enter account ID
documents.azure.com
Please choose an API
* Subscription
Visual Studio Ultimate with MSDN
* Resource Group
Create new
Use existing
* Location
West US
Enable geo-redundancy
Pin to dashboard
Create
Automation options
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
https://portal.azure.com/

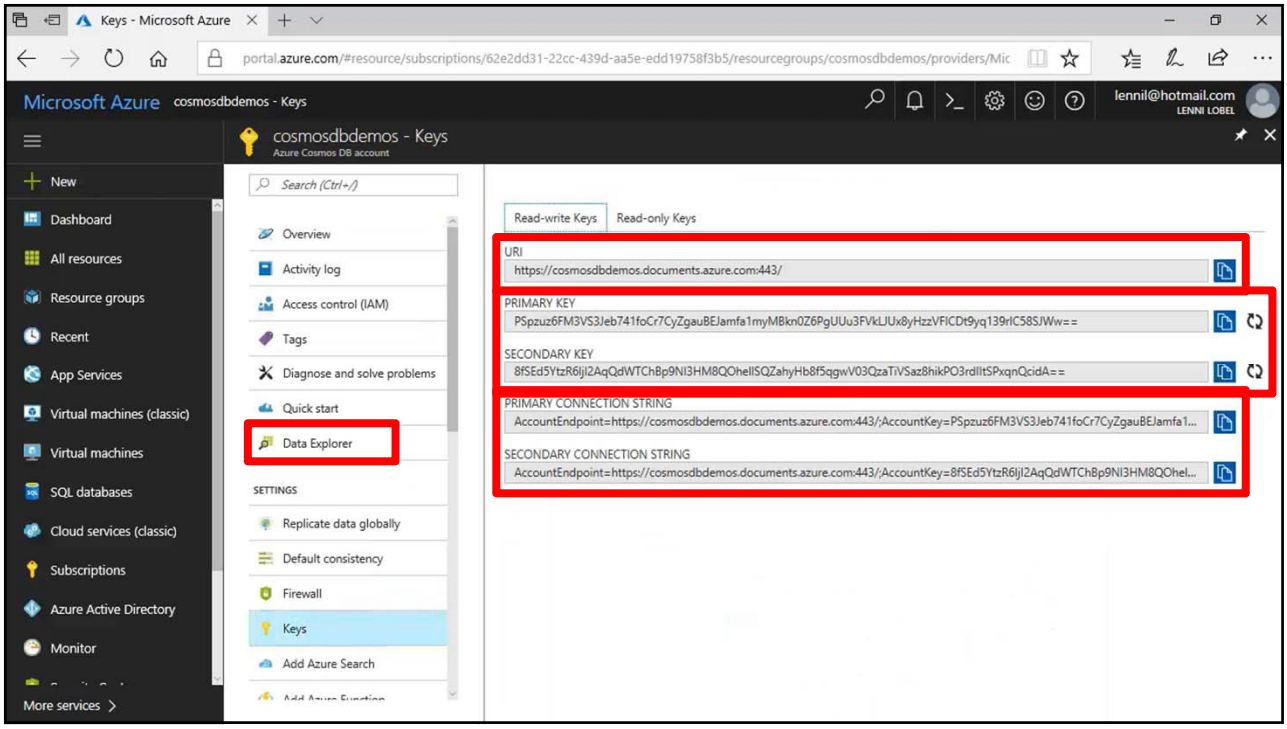

Azure Cosmos DB - Mic
portal.azure.com/#create/Microsoft.DocumentDB
Microsoft Azure
New
Azure Cosmos DB
lennil@hotmail.com
LENNI LOBEL
Azure Cosmos DB
New account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
* ID
cosmosdbdemos
documents.azure.com
Please choose an API
* Subscription
Visual Studio Ultimate with MSDN
* Resource Group
Create new
Use existing
* Location
West US
Enable geo-redundancy
Pin to dashboard
Create
Automation options
https://portal.azure.com/

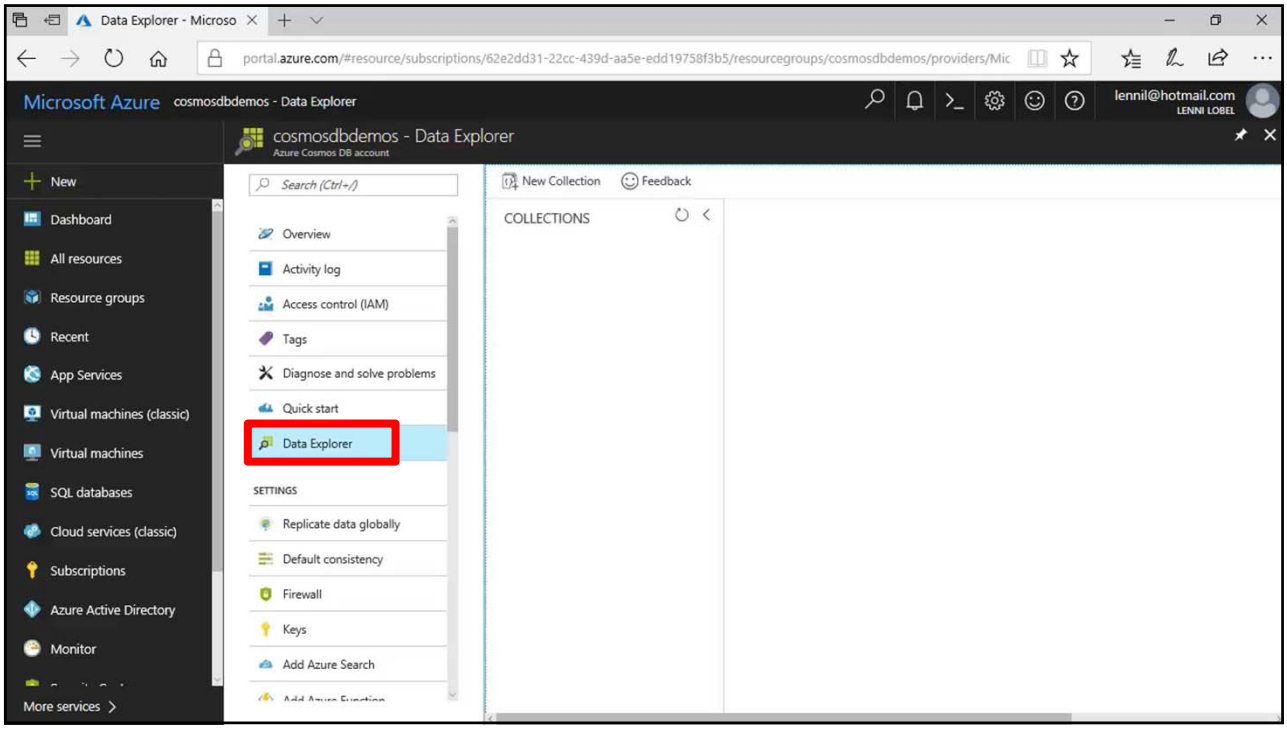

Azure Cosmos DB - Mic
portal.azure.com/#create/Microsoft.DocumentDB
Microsoft Azure
New
Azure Cosmos DB
lennil@hotmail.com
LENNI LOBEL
Azure Cosmos DB
New account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
* ID
cosmosdbdemos
* API
Please choose an API
SQL
MongoDB
Cassandra
Azure Table
Gremlin (graph)
* Location
West US
Enable geo-redundancy
Pin to dashboard
Create
Automation options
More services

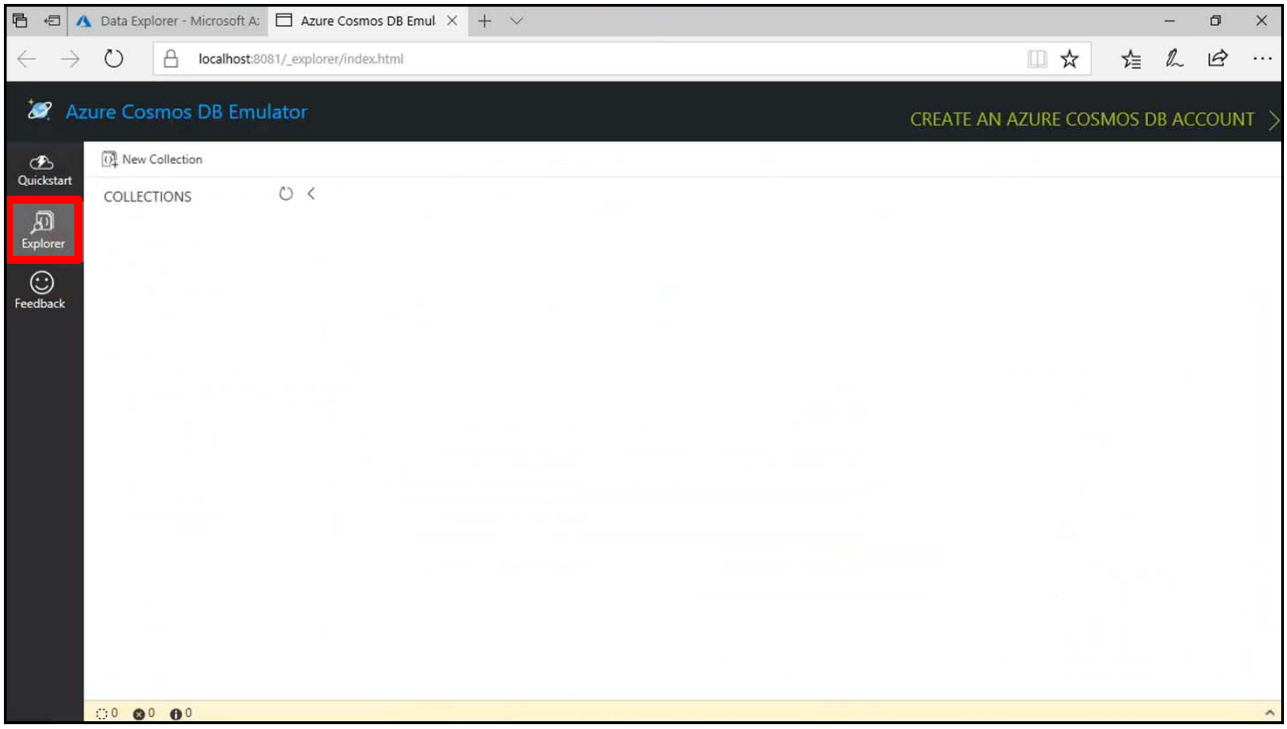

Azure Cosmos DB - Mic
portal.azure.com/#create/Microsoft.DocumentDB
Microsoft Azure
New
Azure Cosmos DB
lennil@hotmail.com
LENNI LOBEL
Azure Cosmos DB
New account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
More services
* ID
cosmosdbdemos
documents.azure.com
* API
SQL
* Subscription
Visual Studio Ultimate with MSDN
* Resource Group
Create new
Use existing
cosmosdbdemos
* Location
West US
Enable geo-redundancy
Pin to dashboard
Create
Automation options


Azure Cosmos DB - Mic
portal.azure.com/#create/Microsoft.DocumentDB
Microsoft Azure
New
Azure Cosmos DB
lennil@hotmail.com
LENNI LOBEL
Azure Cosmos DB
New account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
More services
* ID
cosmosdbdemos
documents.azure.com
* API
SQL
* Subscription
Visual Studio Ultimate with MSDN
* Resource Group
Create new
Use existing
cosmosdbdemos
* Location
West US
Enable geo-redundancy
Enable global distribution on your account by pairing West US with East US. You can add more regions to your account later.
Click here to learn more about global distribution in Azure Cosmos DB
Pin to dashboard
Create
Automation options


Azure Cosmos DB - Mic
portal.azure.com/#resource/subscriptions/62e2dd31-22cc-439d-aa5e-edd19758f3b5/resourcegroups/cosmosdbdemos/providers/Mic
Microsoft Azure
cosmosdbdemos
lennil@hotmail.com
LENNI LOBEL
cosmosdbdemos
Azure Cosmos DB account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
More services
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Quick start
Data Explorer
SETTINGS
Replicate data globally
Default consistency
Firewall
Keys
Add Azure Search
Add Collection
Refresh
Move
Delete Account
Data Explorer
Resource group (change)
cosmosdbdemos
Subscription (change)
Visual Studio Ultimate with MSDN
Read Locations
West US
Write Location
West US
Status
Online
Subscription ID
62e2dd31-22cc-439d-aa5e-edd19758f3b5
URI
https://cosmosdbdemos.documents.azure.com:443/
Collections
Looks like you don't have any collections yet.
Data Explorer
Regions
Region Configuration
COSMOSDBDEMOS


Keys - Microsoft Azure
portal.azure.com/#resource/subscriptions/62e2dd31-22cc-439d-aa5e-edd19758f3b5/resourcegroups/cosmosdbdemos/providers/Mic
Microsoft Azure
cosmosdbdemos - Keys
lennil@hotmail.com
LENNI LOBEL
cosmosdbdemos - Keys
Azure Cosmos DB account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
More services
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Quick start
Data Explorer
SETTINGS
Replicate data globally
Default consistency
Firewall
Keys
Add Azure Search
Read-write Keys
Read-only Keys
URI
https://cosmosdbdemos.documents.azure.com:443/
PRIMARY KEY
PSpzuz6FM3VS3Jeb741foCr7CyZgauBEJamfa1myMBkn0Z6PgUUu3FVkLJUx8yHzzVFICDt9yq139rlC58SJWw==
SECONDARY KEY
8fSEd5YtzR6ljI2AqQdWTChBp9NI3HM8QOhelISQZahyHb8f5qgwV03QzaTiVSaz8hikPO3rdlItSPxqnQcidA==
PRIMARY CONNECTION STRING
AccountEndpoint=https://cosmosdbdemos.documents.azure.com:443/;AccountKey=PSpzuz6FM3VS3Jeb741foCr7CyZgauBEJamfa1...
SECONDARY CONNECTION STRING
AccountEndpoint=https://cosmosdbdemos.documents.azure.com:443/;AccountKey=8fSEd5YtzR6ljI2AqQdWTChBp9NI3HM8QOhel...


Data Explorer - Microso
portal.azure.com/#resource/subscriptions/62e2dd31-22cc-439d-aa5e-edd19758f3b5/resourcegroups/cosmosdbdemos/providers/Mic
Microsoft Azure
cosmosdbdemos - Data Explorer
lennil@hotmail.com
LENNI LOBEL
cosmosdbdemos - Data Explorer
Azure Cosmos DB account
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Azure Active Directory
Monitor
More services
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Quick start
Data Explorer
SETTINGS
Replicate data globally
Default consistency
Firewall
Keys
Add Azure Search
New Collection
Feedback
COLLECTIONS


Data Explorer - Microsoft A
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS

Introducing the Local Emulator

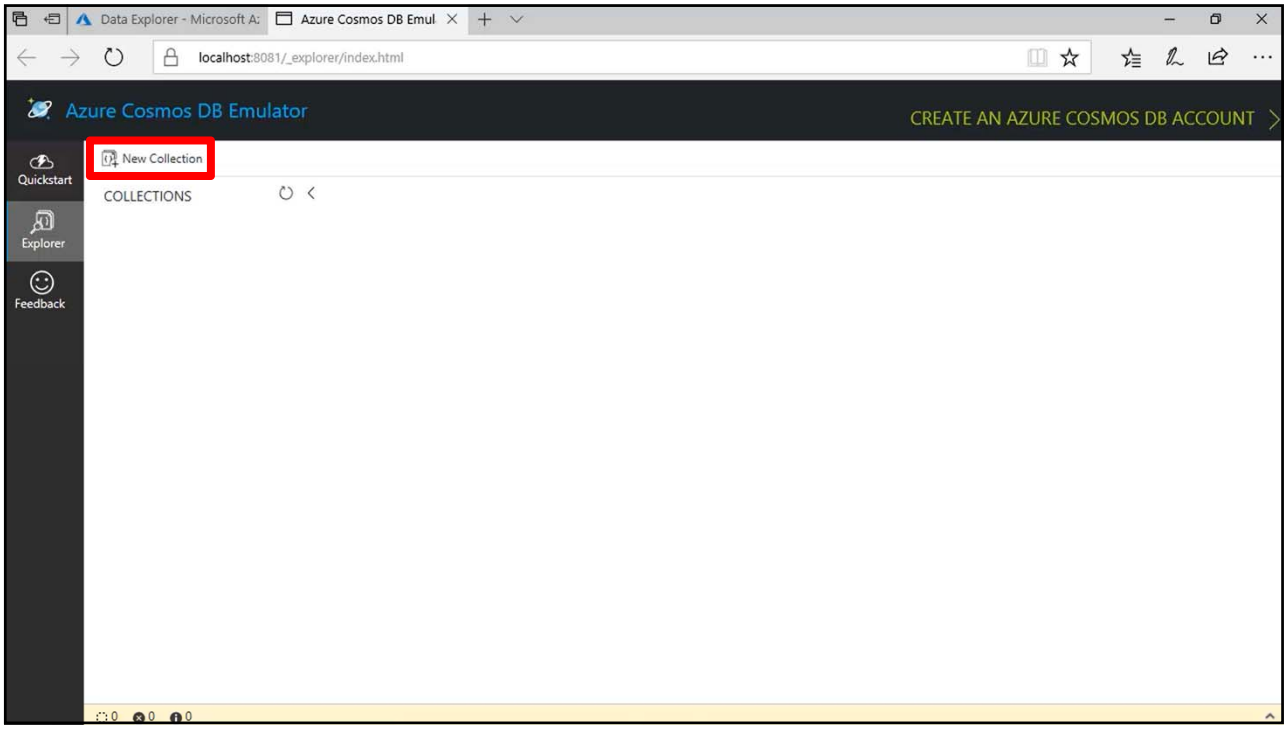
- Emulate Cosmos DB in a local development environment
  - Supports identical functionality as Azure Cosmos DB in the cloud

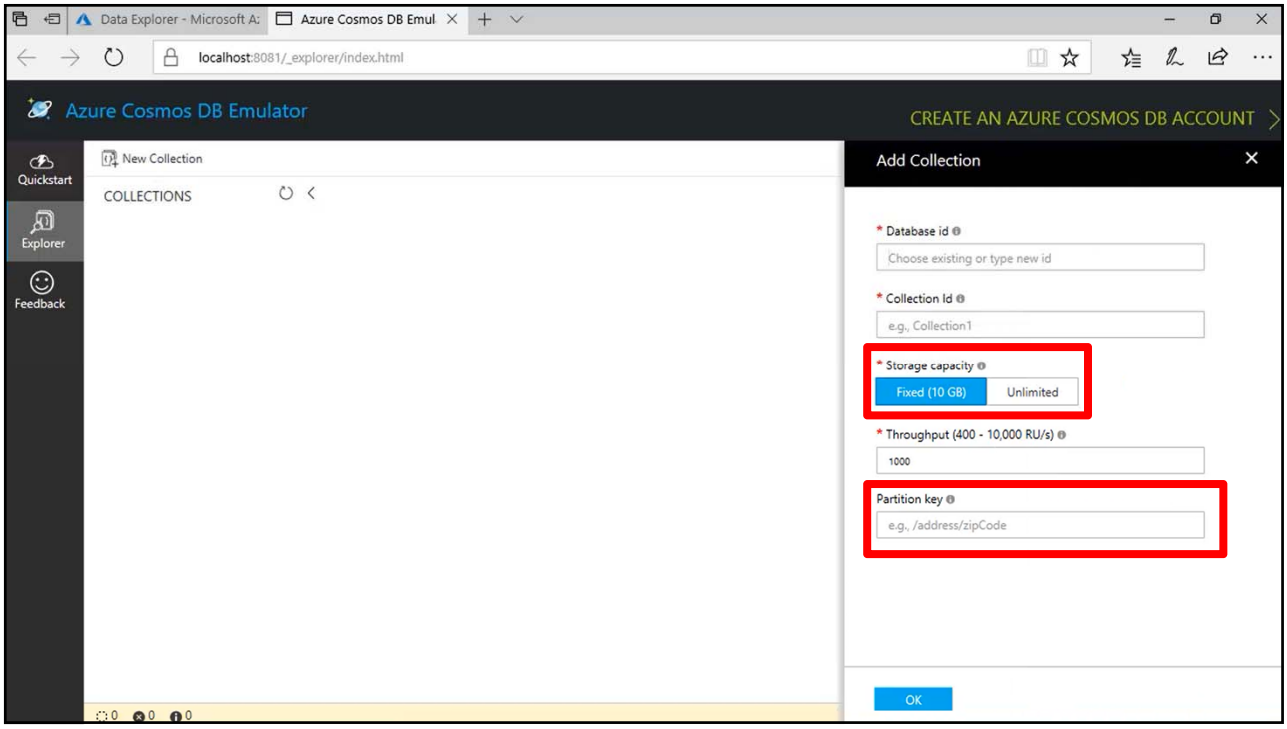
- No need for:
  - Azure subscription
  - Cosmos DB account
  - Internet connection

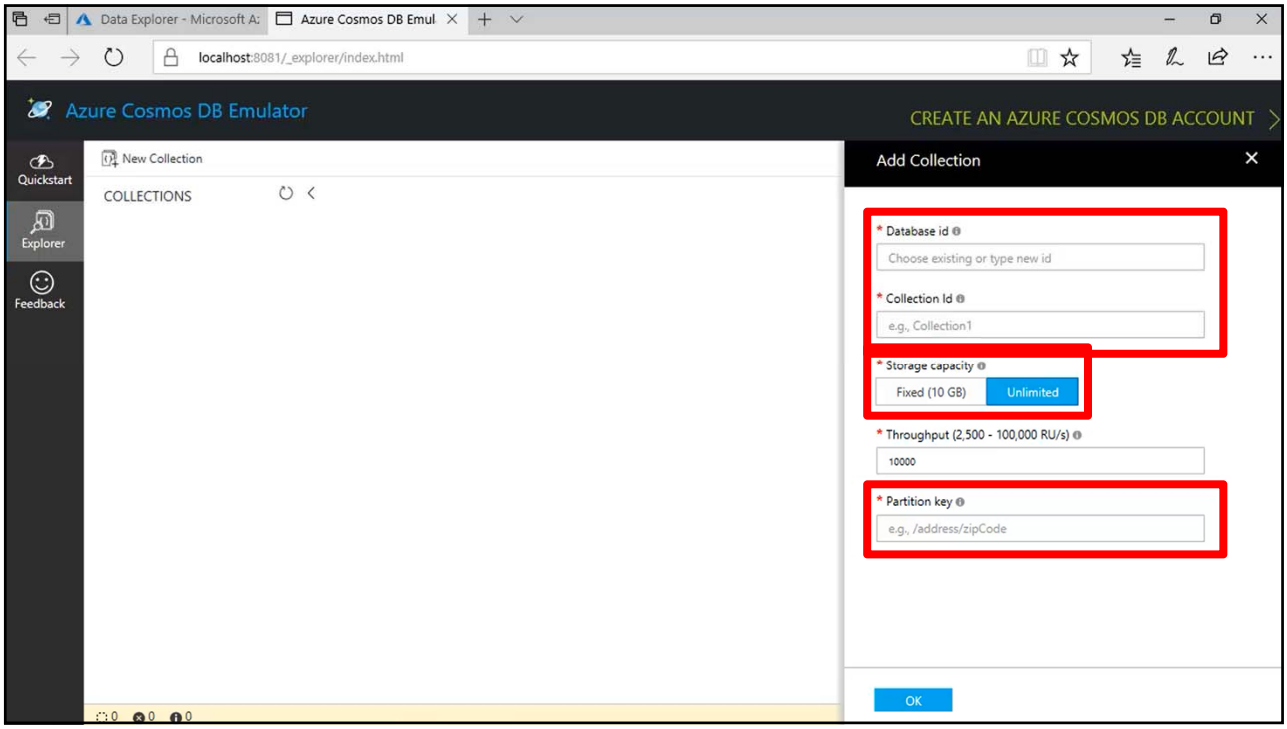
- Develop and test locally
  - Incur no costs
  - Deploy to the cloud when ready

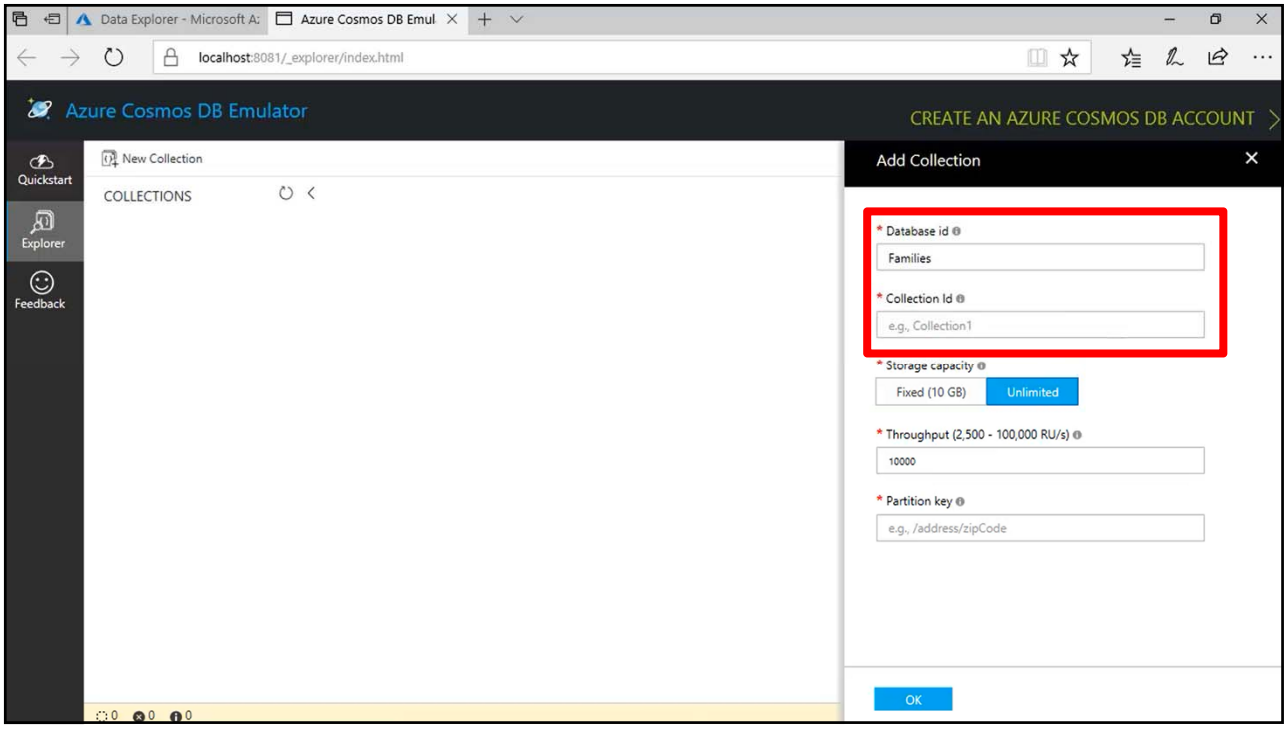
Demo

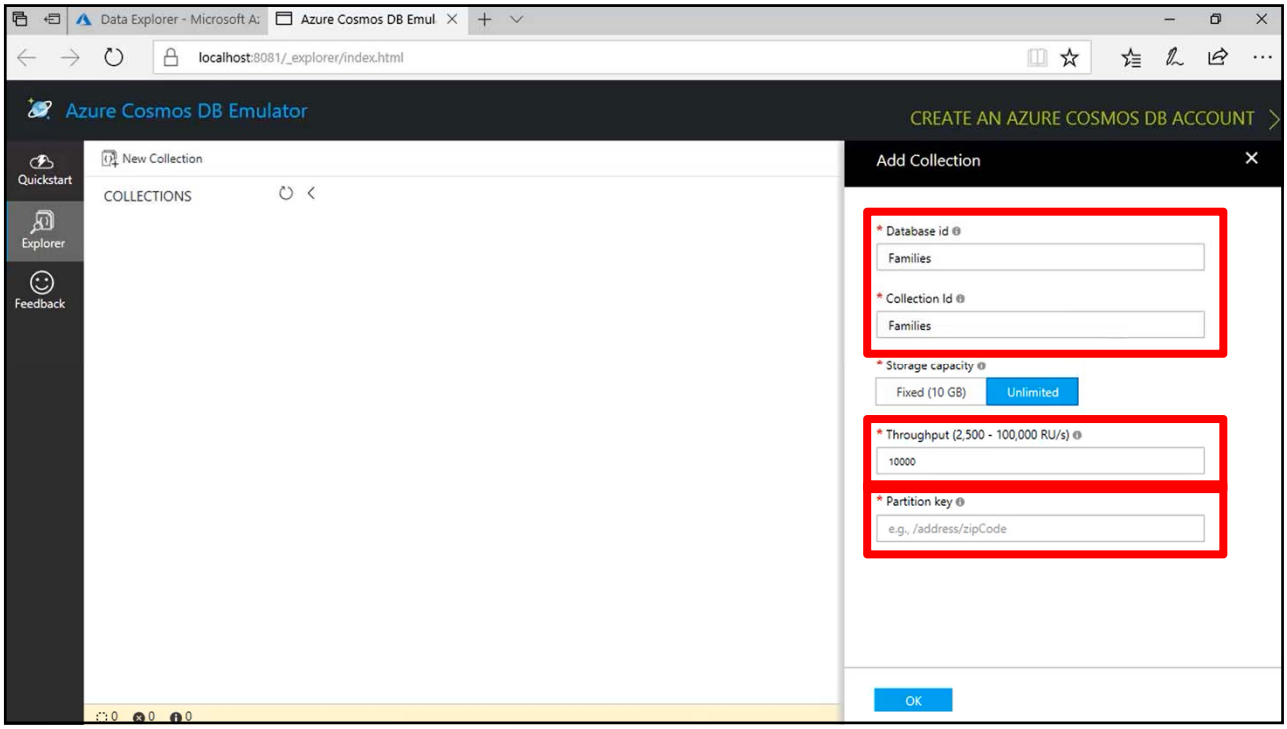
Creating a collection

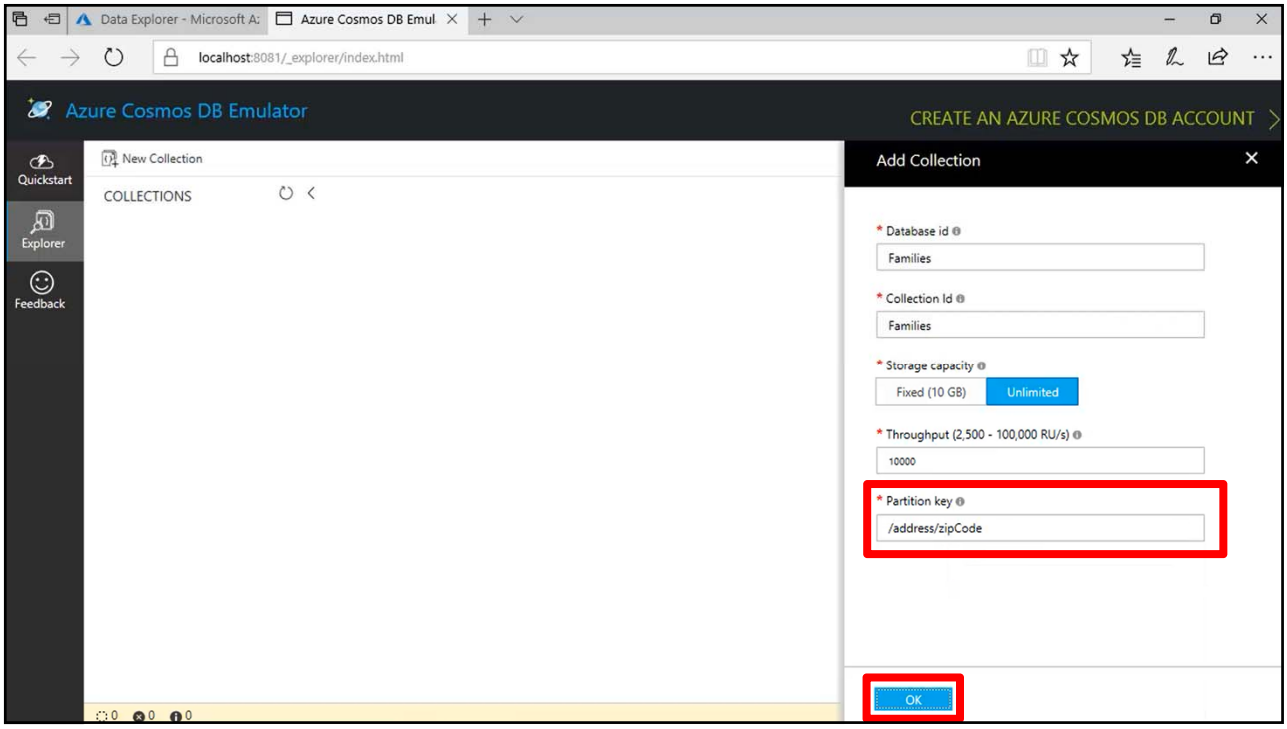

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS

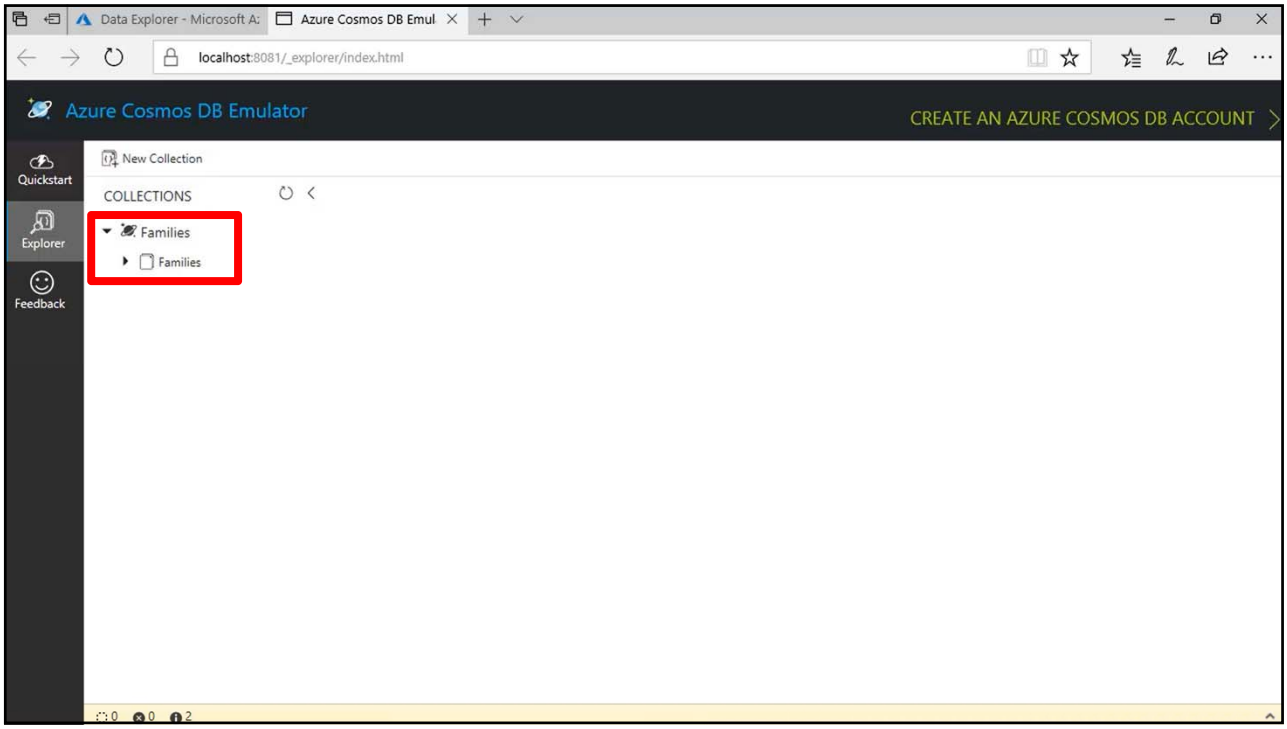

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS
Add Collection
* Database id
Choose existing or type new id
* Collection Id
e.g., Collection1
* Storage capacity
Fixed (10 GB)
Unlimited
* Throughput (400 - 10,000 RU/s)
1000
Partition key
e.g., /address/zipCode
OK

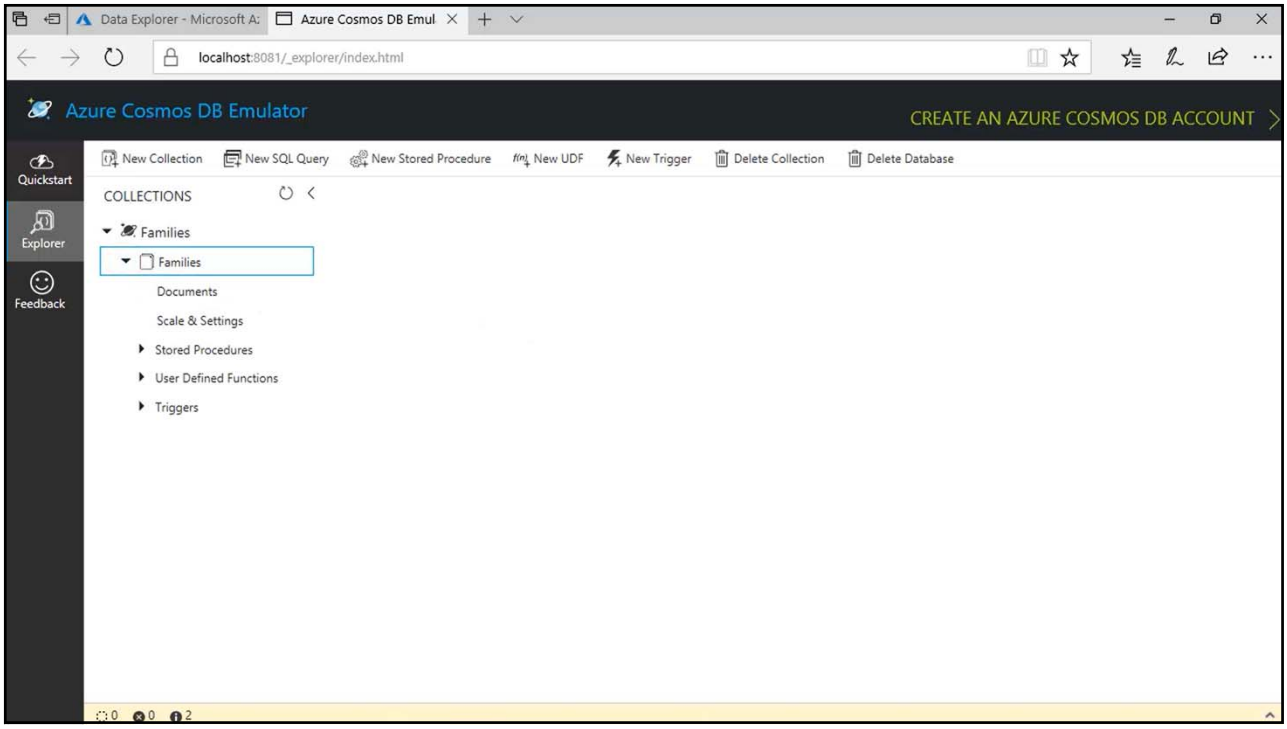

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS
Add Collection
* Database id
Choose existing or type new id
* Collection Id
e.g., Collection1
* Storage capacity
Fixed (10 GB)
Unlimited
* Throughput (2,500 - 100,000 RU/s)
10000
* Partition key
e.g., /address/zipCode
OK


Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS
Add Collection
* Database id
Families
* Collection Id
e.g., Collection1
* Storage capacity
Fixed (10 GB)
Unlimited
* Throughput (2,500 - 100,000 RU/s)
10000
* Partition key
e.g., /address/zipCode
OK


Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS
Add Collection
* Database id
Families
* Collection Id
Families
* Storage capacity
Fixed (10 GB)
Unlimited
* Throughput (2,500 - 100,000 RU/s)
10000
* Partition key
e.g., /address/zipCode
OK

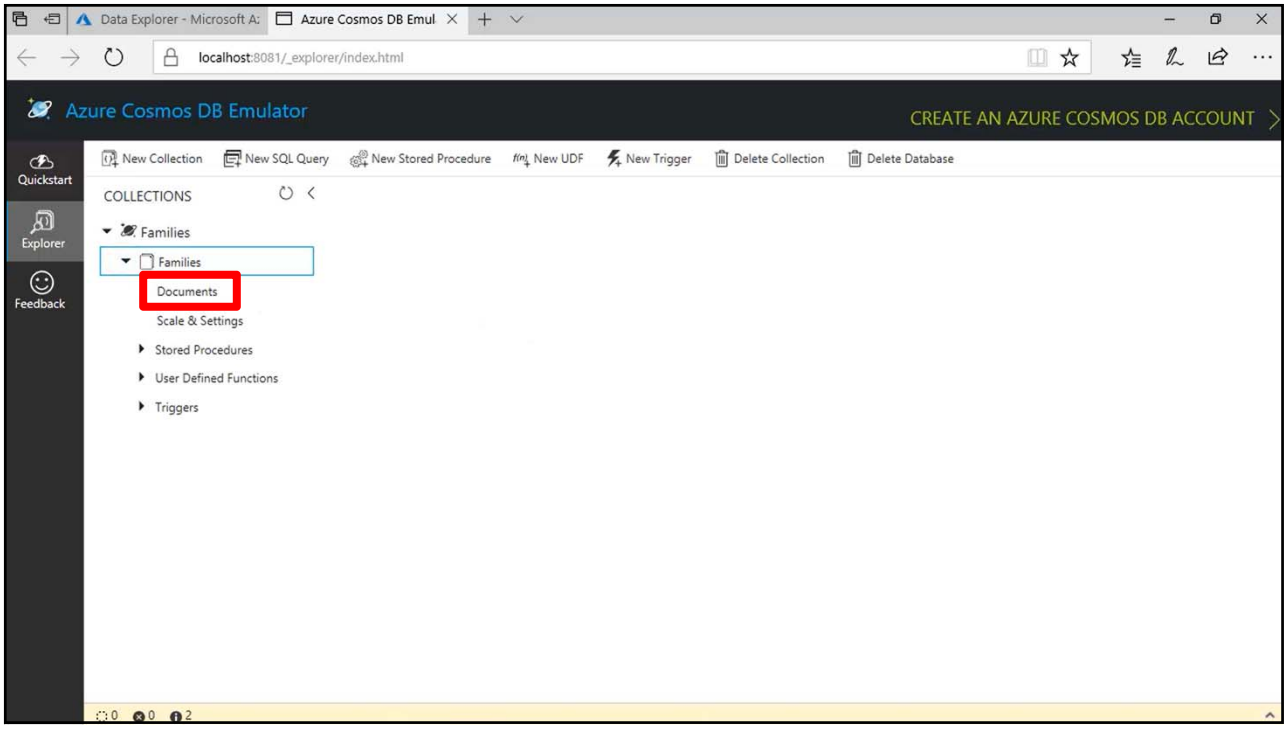

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS
Add Collection
* Database id
Families
* Collection Id
Families
* Storage capacity
Fixed (10 GB)
Unlimited
* Throughput (2,500 - 100,000 RU/s)
10000
* Partition key
/address/zipCode
OK

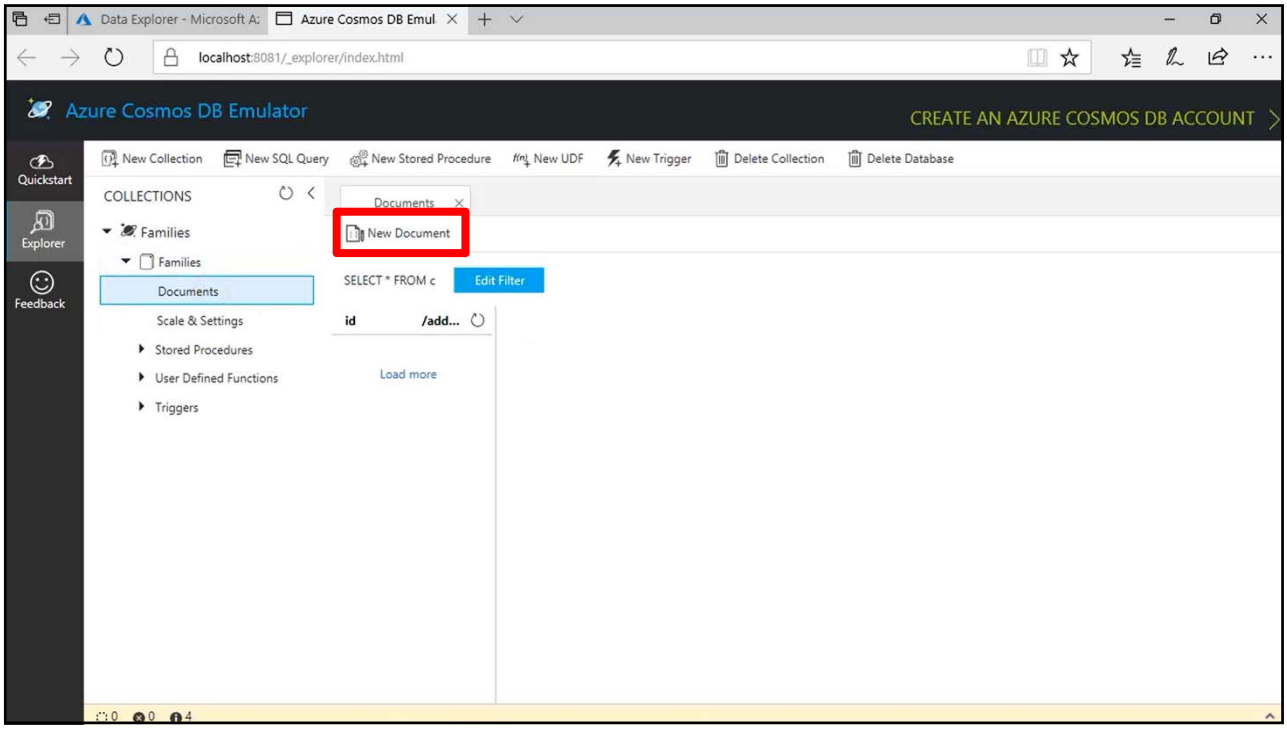

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
COLLECTIONS
Families
Families

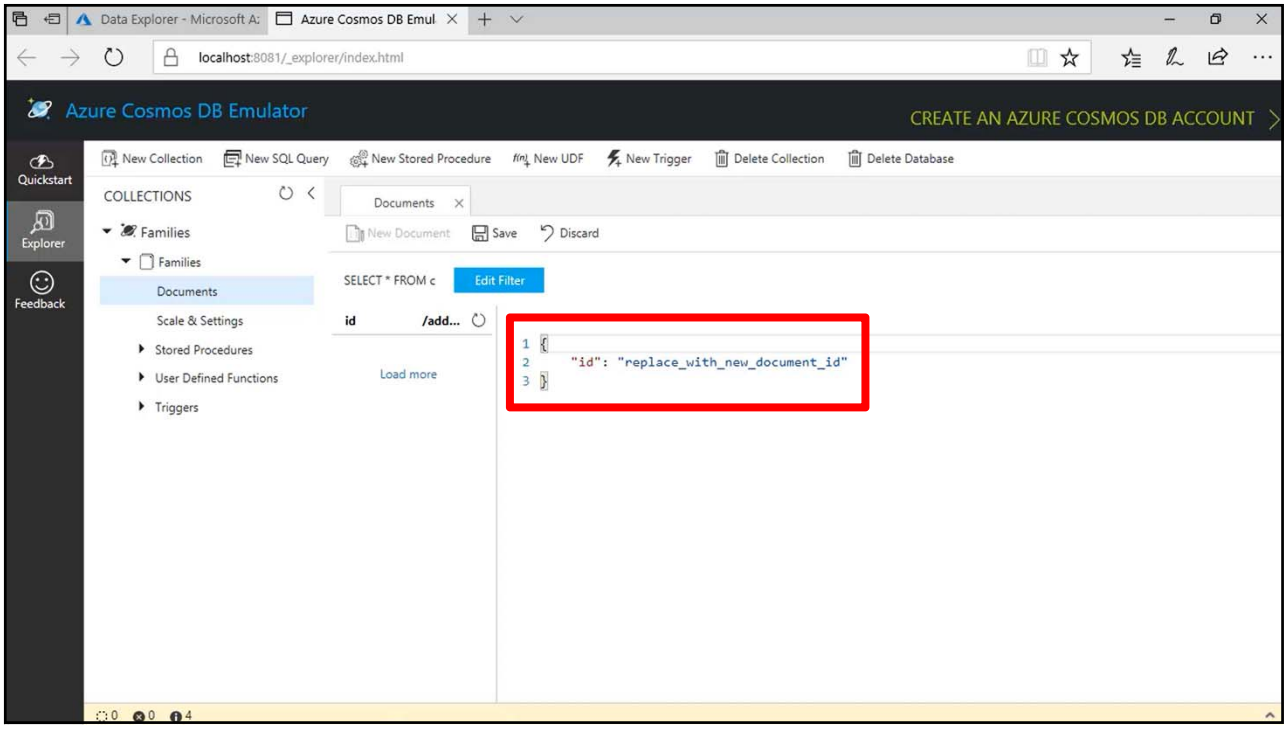

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers

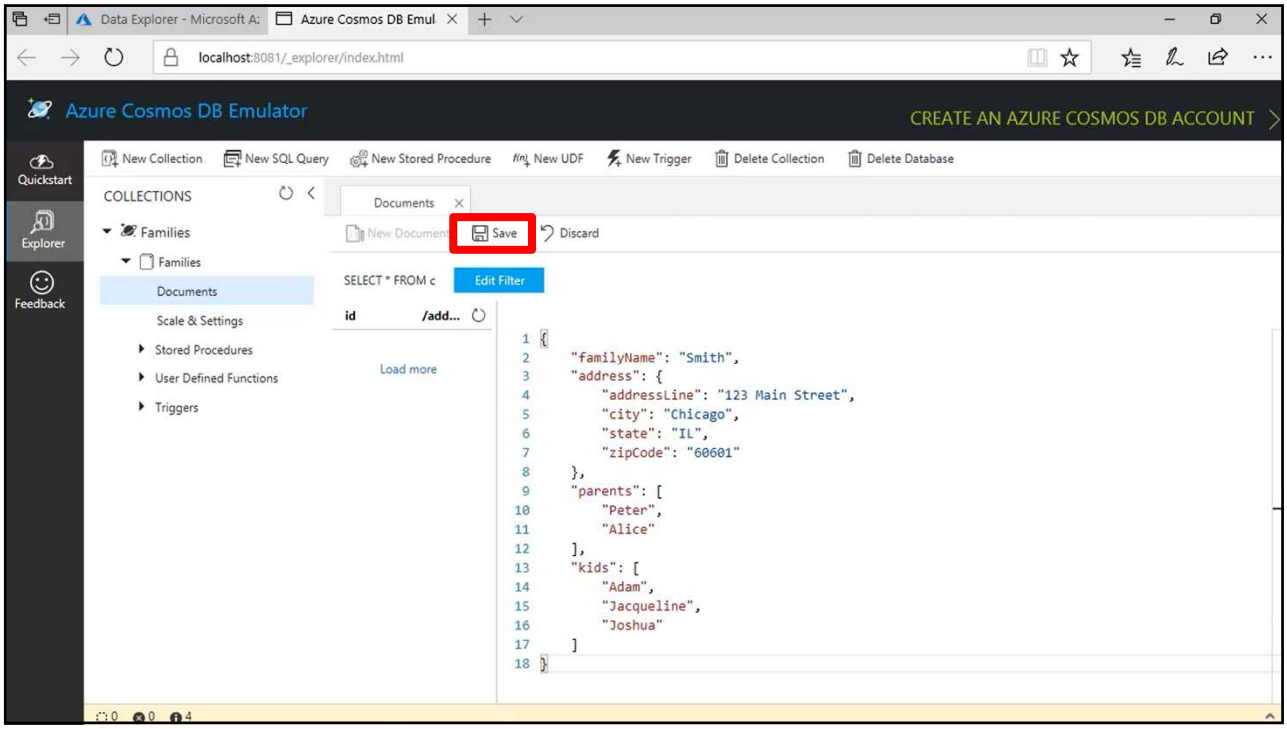
Demo

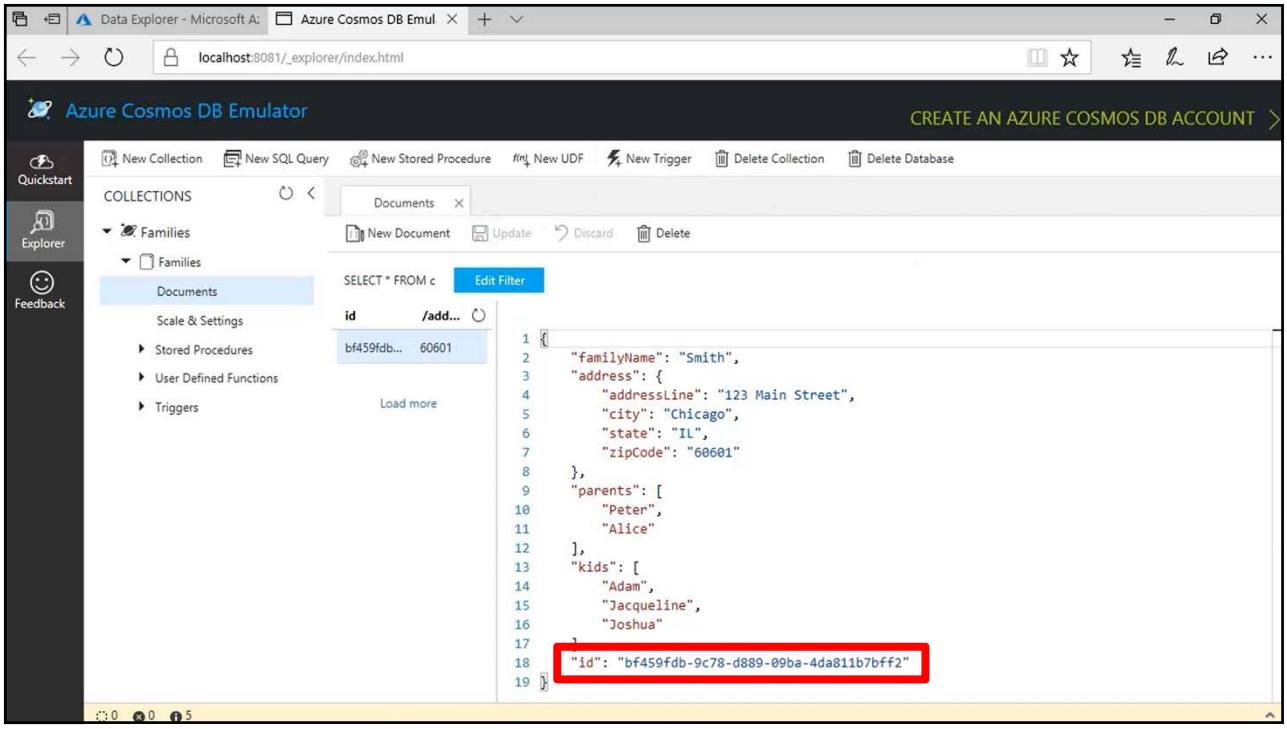
Creating documents

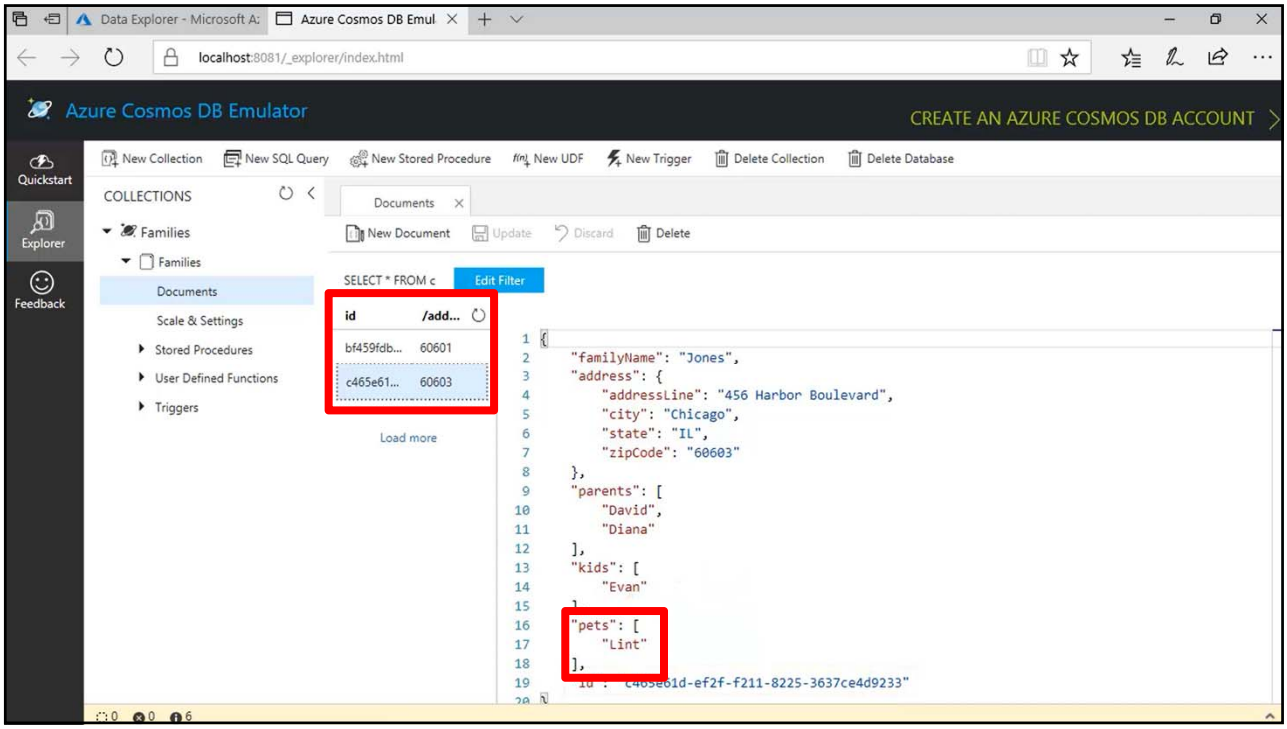


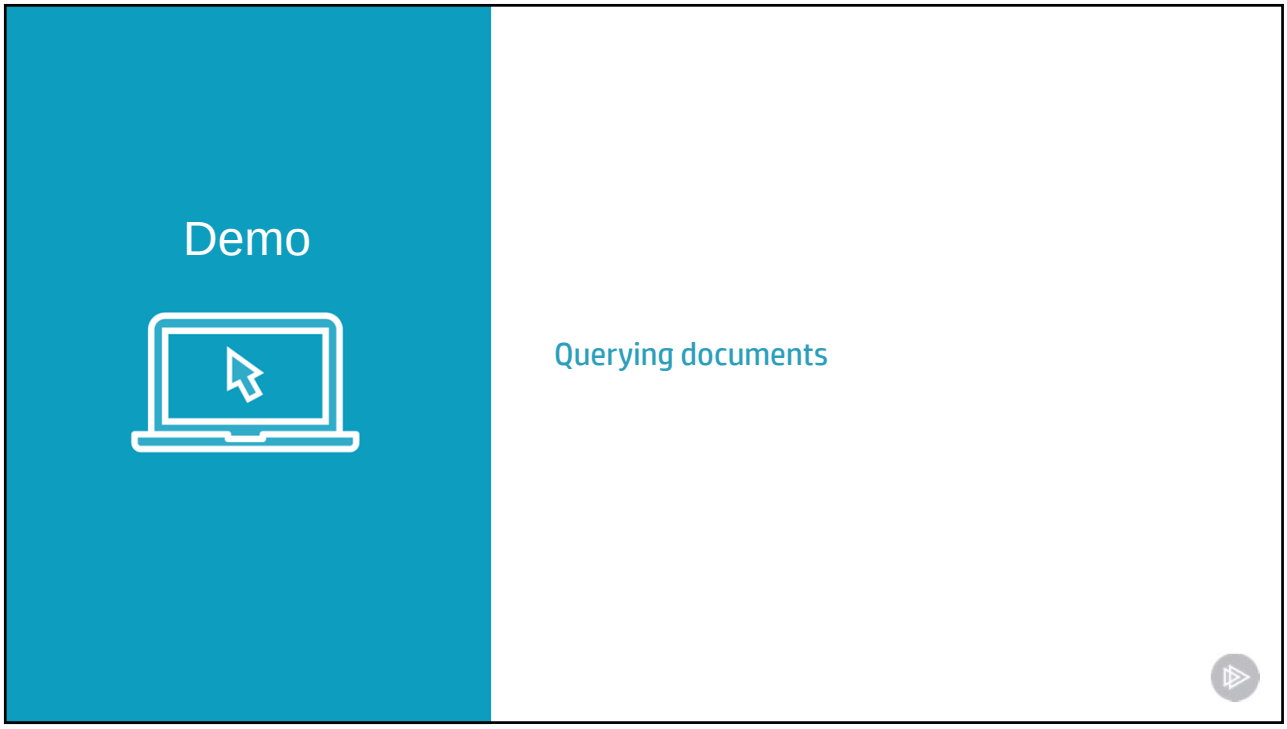

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
New Document
SELECT * FROM c
Edit Filter
id
/add...
Load more

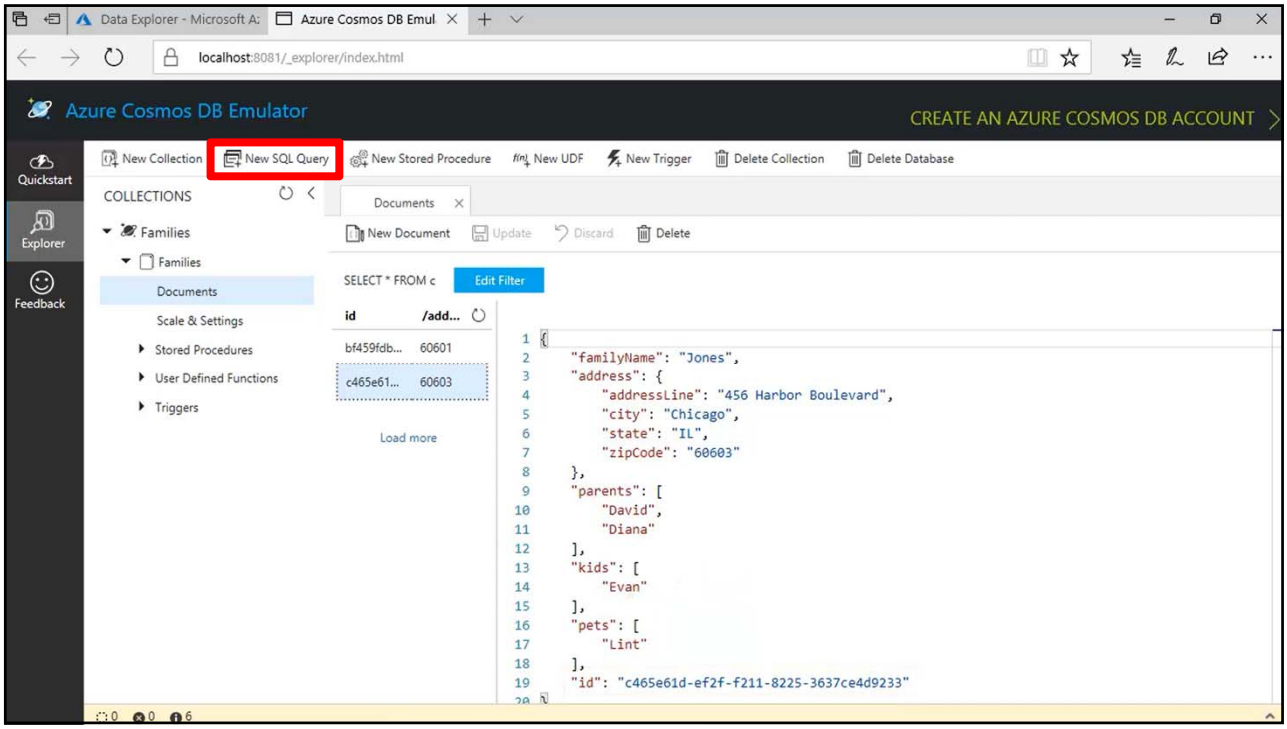

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
New Document
Save
Discard
SELECT * FROM c
Edit Filter
id
/add...
Load more
{
"id": "replace_with_new_document_id"
}

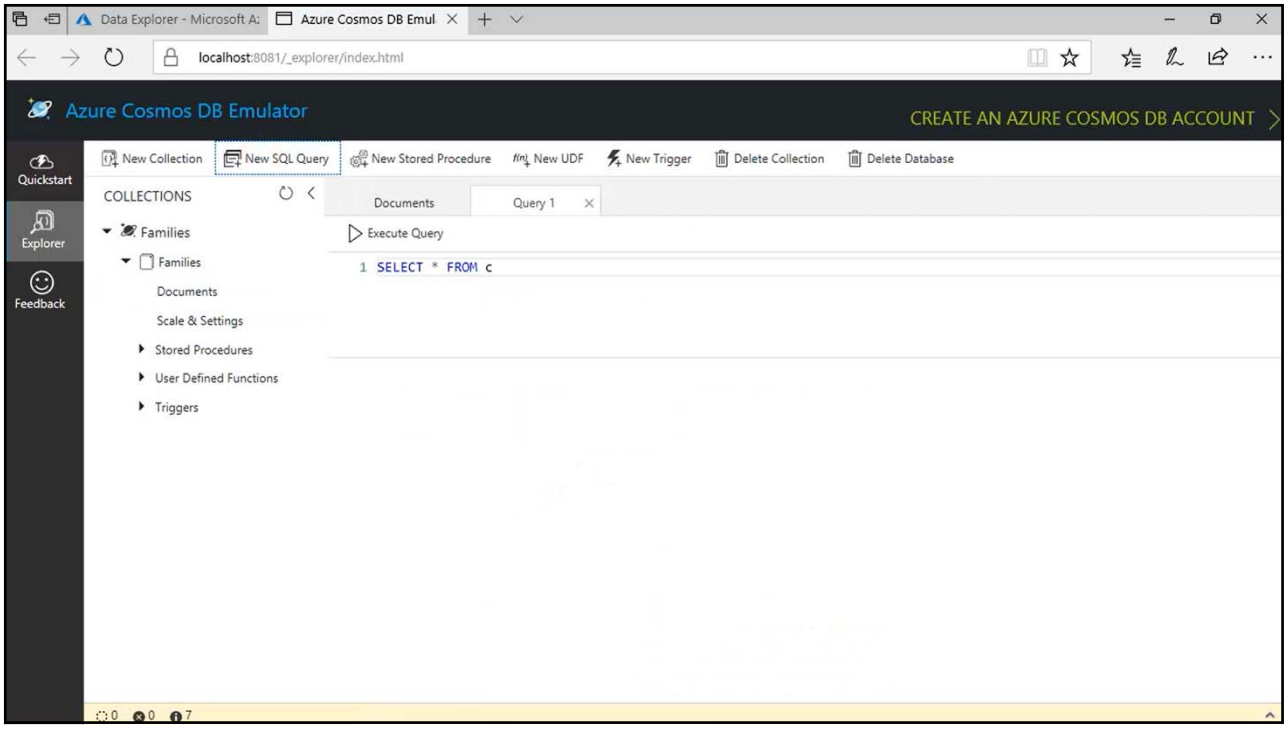

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Save
Discard
SELECT * FROM c
Edit Filter
id
/add...
Load more
{
"familyName": "Smith",
"address": {
"addressLine": "123 Main Street",
"city": "Chicago",
"state": "IL",
"zipCode": "60601"
},
"parents": [
"Peter",
"Alice"
],
"kids": [
"Adam",
"Jacqueline",
"Joshua"
]
}

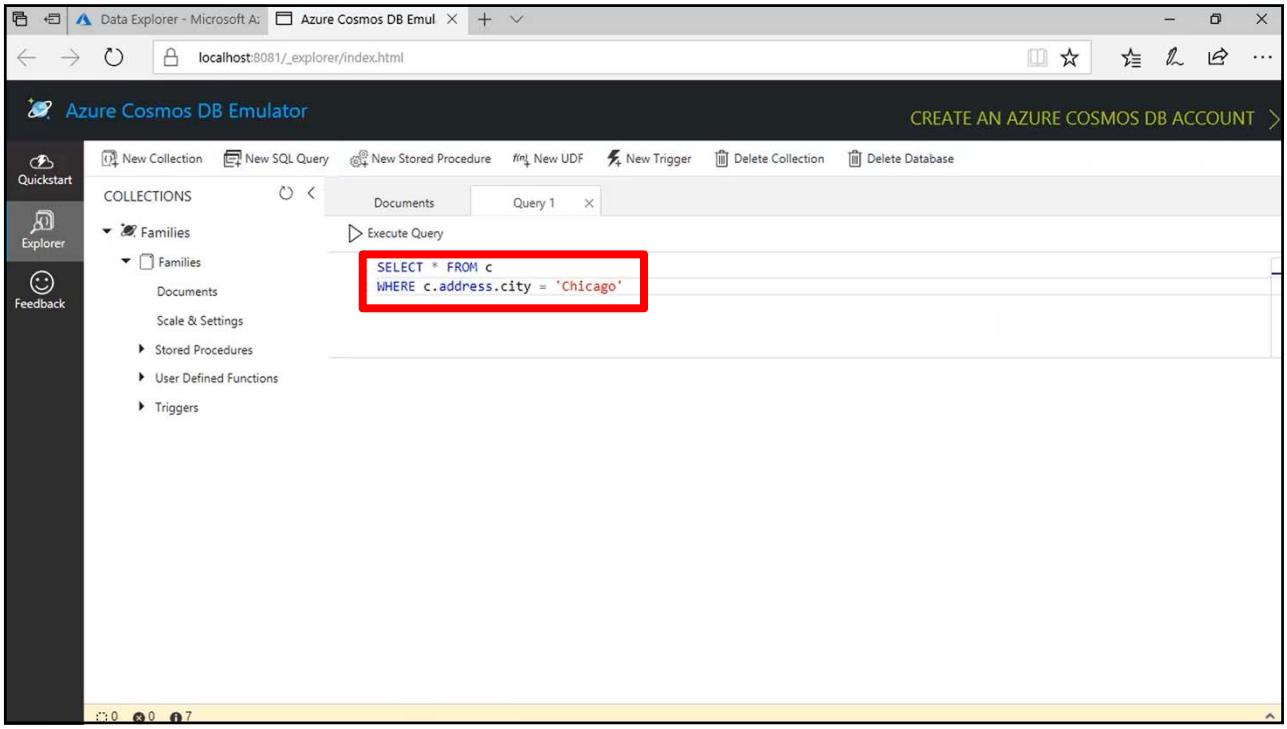

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
New Document
Update
Discard
Delete
SELECT * FROM c
Edit Filter
id
/add...
bf459fdb...
60601
Load more
{
"familyName": "Smith",
"address": {
"addressLine": "123 Main Street",
"city": "Chicago",
"state": "IL",
"zipCode": "60601"
},
"parents": [
"Peter",
"Alice"
],
"kids": [
"Adam",
"Jacqueline",
"Joshua"
],
"id": "bf459fdb-9c78-d889-09ba-4da811b7bff2"
}

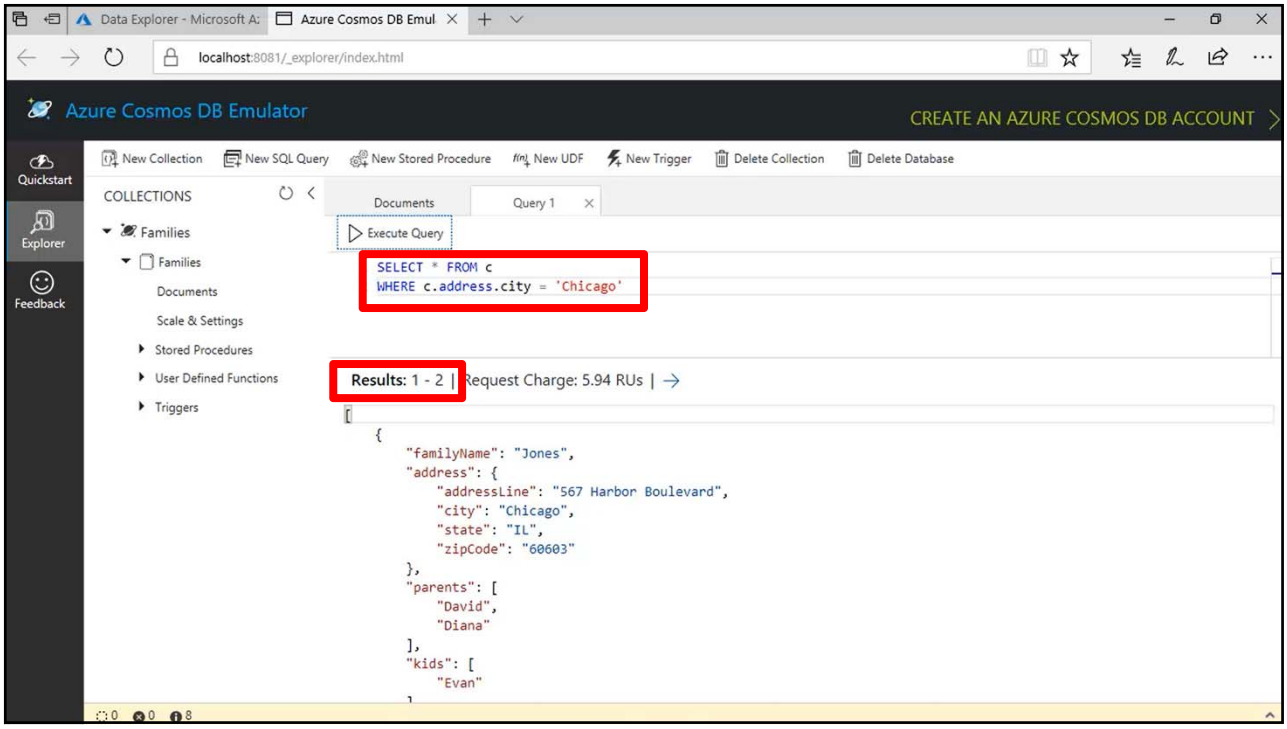

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
New Document
Update
Discard
Delete
SELECT * FROM c
Edit Filter
id
/add...
bf459fdb...
60601
c465e61...
60603
Load more
{
"familyName": "Jones",
"address": {
"addressLine": "456 Harbor Boulevard",
"city": "Chicago",
"state": "IL",
"zipCode": "60603"
},
"parents": [
"David",
"Diana"
],
"kids": [
"Evan"
],
"pets": [
"Lint"
],
"id": "c465e61d-ef2f-f211-8225-3637ce4d9233"
}

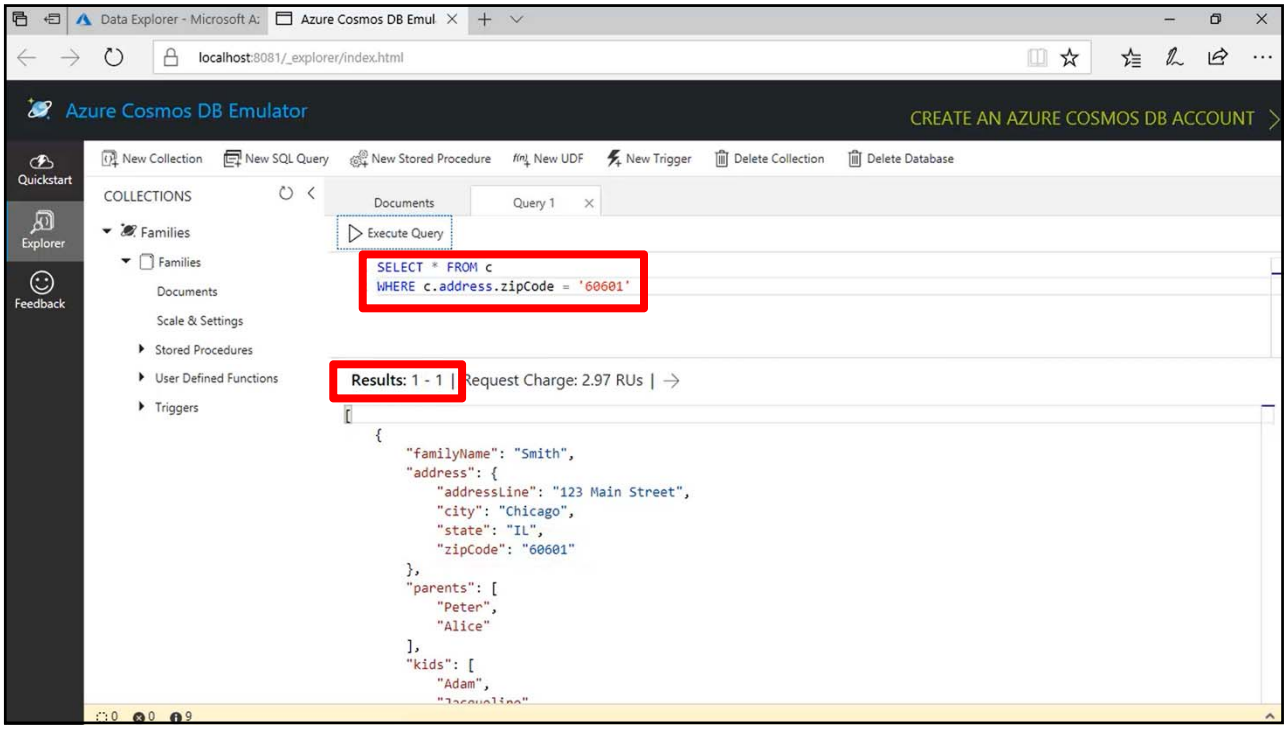

Demo
Querying documents

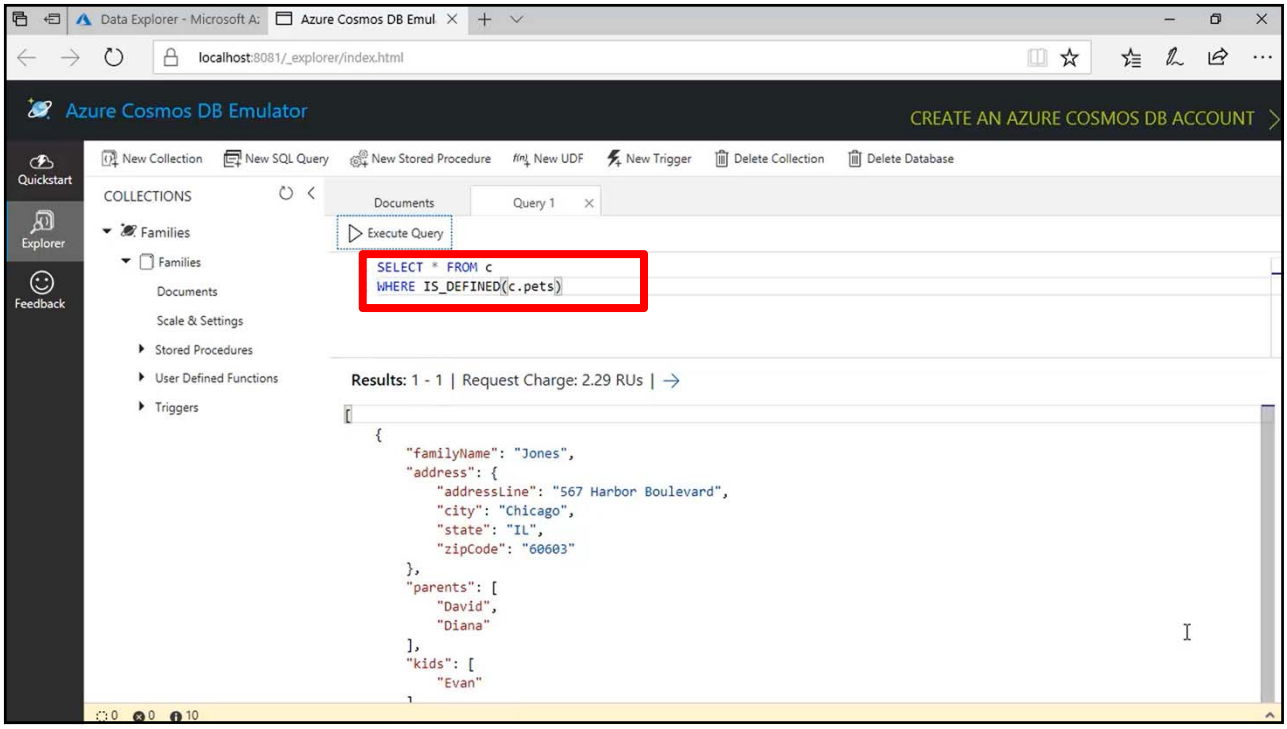

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
New Document
Update
Discard
Delete
SELECT * FROM c
Edit Filter
id
/add...
bf459fdb...
60601
c465e61...
60603
Load more
{
"familyName": "Jones",
"address": {
"addressLine": "456 Harbor Boulevard",
"city": "Chicago",
"state": "IL",
"zipCode": "60603"
},
"parents": [
"David",
"Diana"
],
"kids": [
"Evan"
],
"pets": [
"Lint"
],
"id": "c465e61d-ef2f-f211-8225-3637ce4d9233"
}

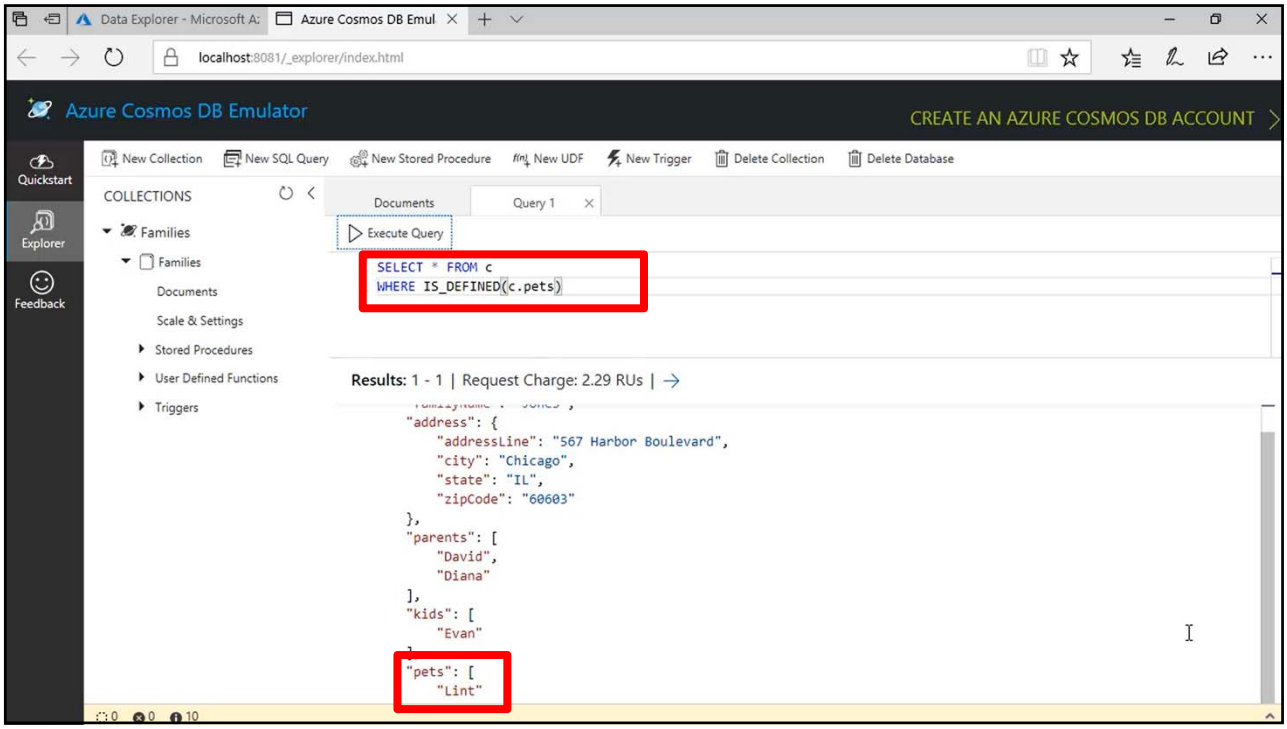

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c

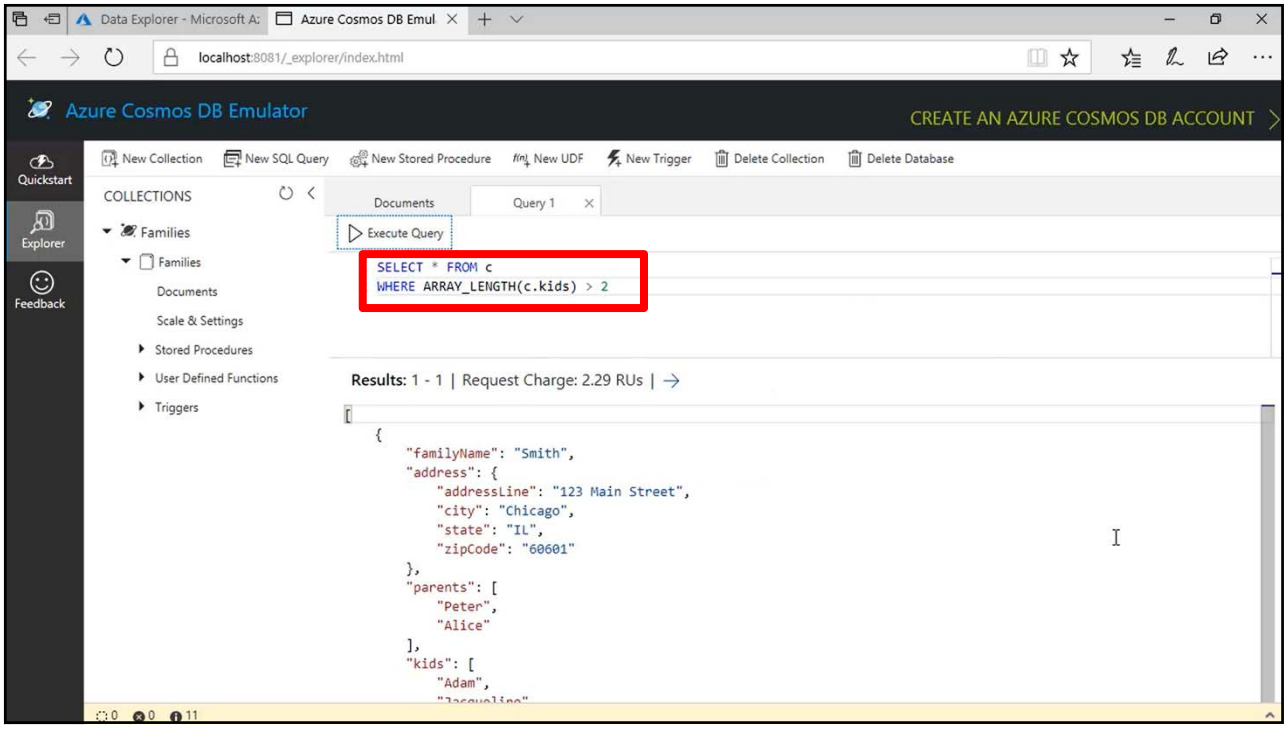

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c
WHERE c.address.city = 'Chicago'

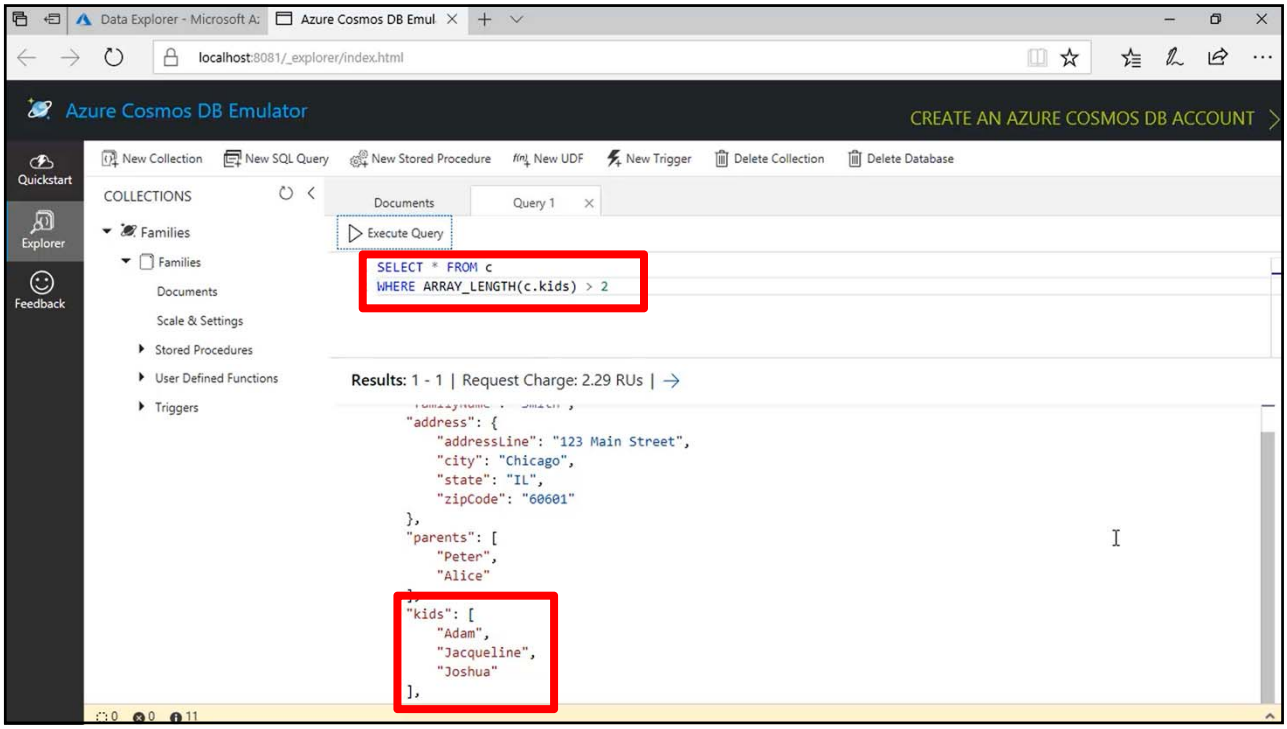

Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c
WHERE c.address.city = 'Chicago'
Results: 1 - 2 | Request Charge: 5.94 RUs |
{
"familyName": "Jones",
"address": {
"addressLine": "567 Harbor Boulevard",
"city": "Chicago",
"state": "IL",
"zipCode": "60603"
},
"parents": [
"David",
"Diana"
],
"kids": [
"Evan"


Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c
WHERE c.address.zipCode = '60601'
Results: 1 - 1 | Request Charge: 2.97 RUs |
{
"familyName": "Smith",
"address": {
"addressLine": "123 Main Street",
"city": "Chicago",
"state": "IL",
"zipCode": "60601"
},
"parents": [
"Peter",
"Alice"
],
"kids": [
"Adam",


Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c
WHERE IS_DEFINED(c.pets)
Results: 1 - 1 | Request Charge: 2.29 RUs |
{
"familyName": "Jones",
"address": {
"addressLine": "567 Harbor Boulevard",
"city": "Chicago",
"state": "IL",
"zipCode": "60603"
},
"parents": [
"David",
"Diana"
],
"kids": [
"Evan"


Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c
WHERE IS_DEFINED(c.pets)
Results: 1 - 1 | Request Charge: 2.29 RUs
"address": {
"addressLine": "567 Harbor Boulevard",
"city": "Chicago",
"state": "IL",
"zipCode": "60603"
},
"parents": [
"David",
"Diana"
],
"kids": [
"Evan"
],
"pets": [
"Lint"


Data Explorer - Microsoft A:
Azure Cosmos DB Emul
localhost:8081/_explorer/index.html
Azure Cosmos DB Emulator
CREATE AN AZURE COSMOS DB ACCOUNT
Quickstart
Explorer
Feedback
New Collection
New SQL Query
New Stored Procedure
New UDF
New Trigger
Delete Collection
Delete Database
COLLECTIONS
Families
Documents
Scale & Settings
Stored Procedures
User Defined Functions
Triggers
Documents
Query 1
Execute Query
SELECT * FROM c
WHERE ARRAY_LENGTH(c.kids) > 2
Results: 1 - 1 | Request Charge: 2.29 RUs
[
{
"familyName": "Smith",
"address": {
"addressLine": "123 Main Street",
"city": "Chicago",
"state": "IL",
"zipCode": "60601"
},
"parents": [
"Peter",
"Alice"
],
"kids": [
"Adam",



Multiple APIs and Data Models

Document

SQL API (JSON)
MongoDB API (BSON)

Key-Value

Table API
(replaces Azure Table Storage)

Graph

Gremlin API
(graph traversal language)

Columnar

Cassandra API

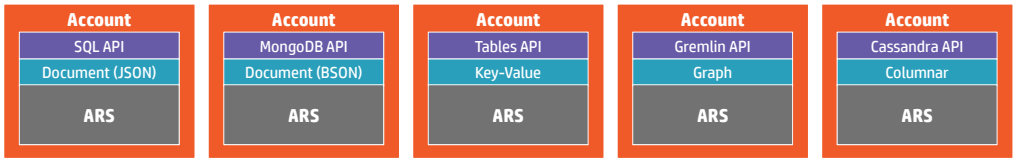
Atom Record Sequence — **ARS**

It's all ARS under the covers,
projected as any desired data model

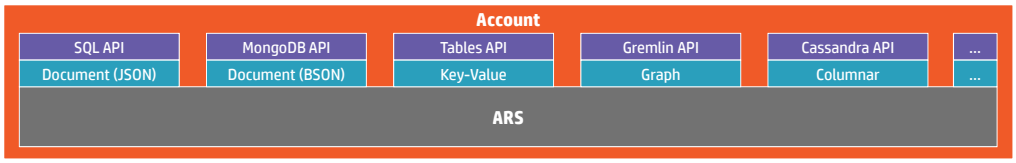
Multiple APIs and Data Models

API
Data Model

- Today



- In the future



Throughput and Performance

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 (CPU, memory, disk I/O, network I/O)

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

Reserving Request Units

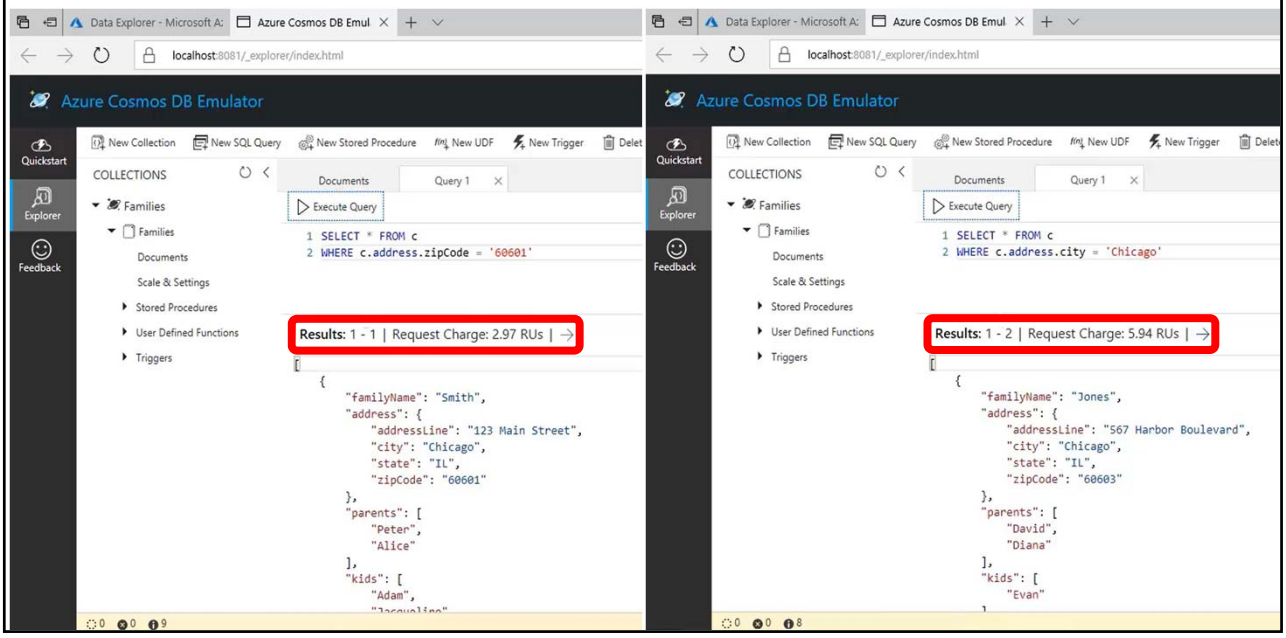
Provision request units per second (RU/s)

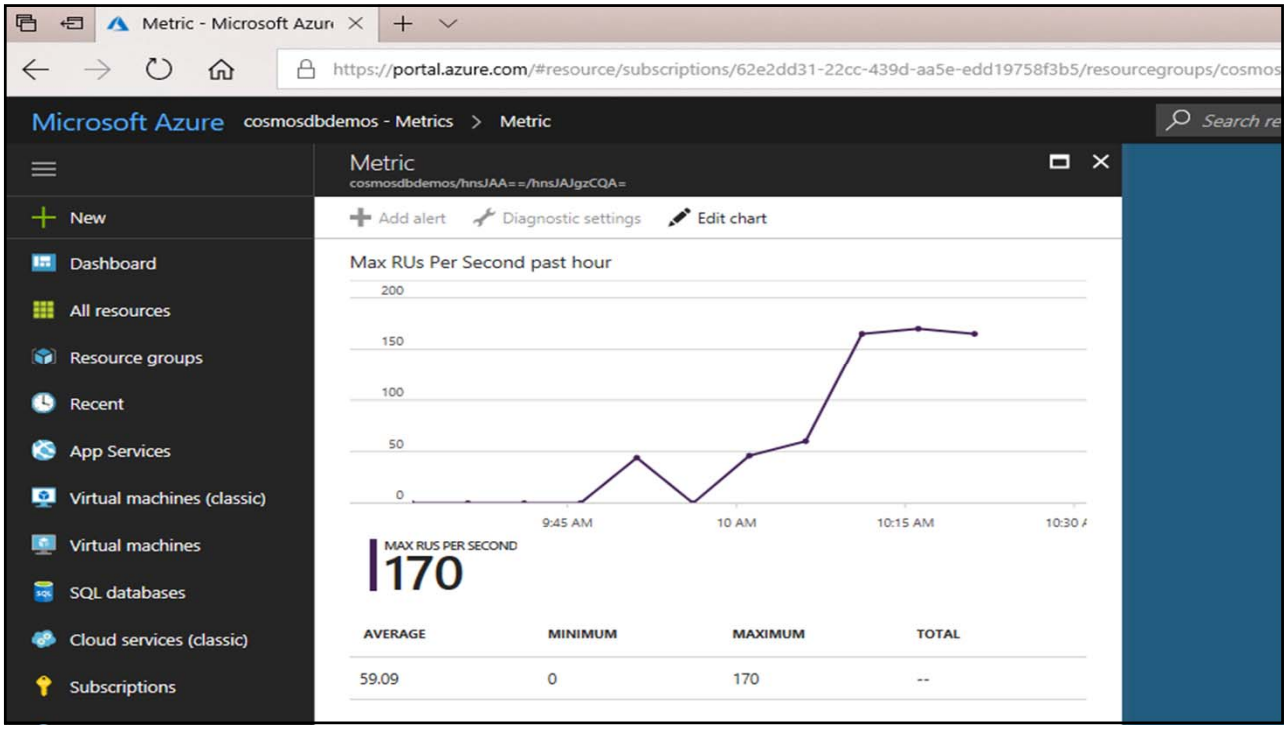
How many request *units* (not *requests*) per second are available to your application

Exceeding reserved throughput limits

Requests are "throttled"

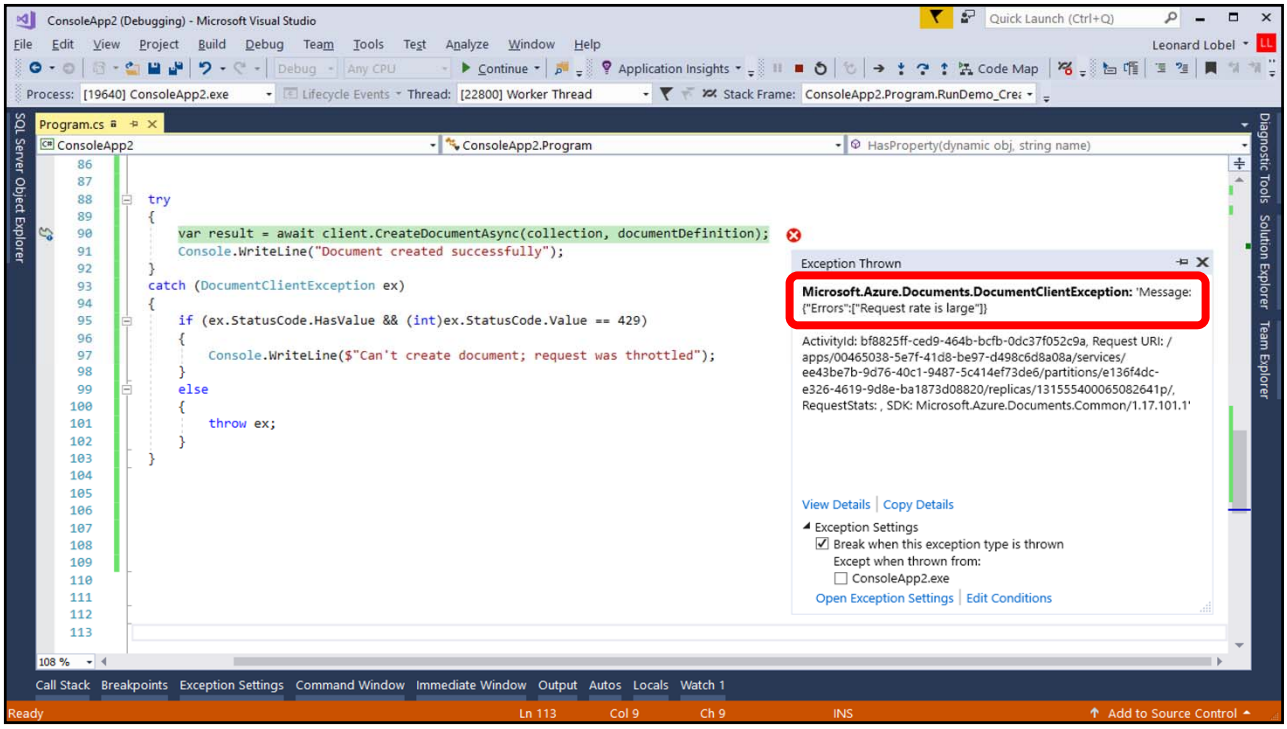
Monitoring Request Unit Consumption

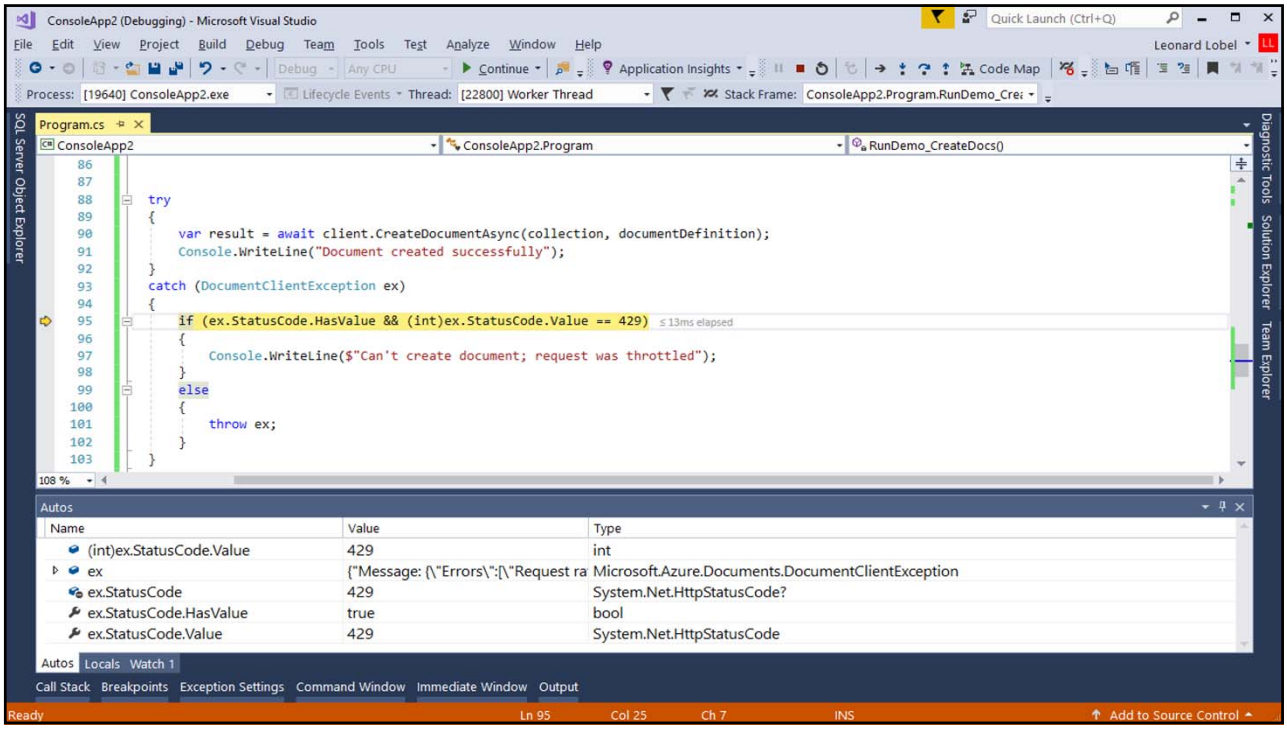


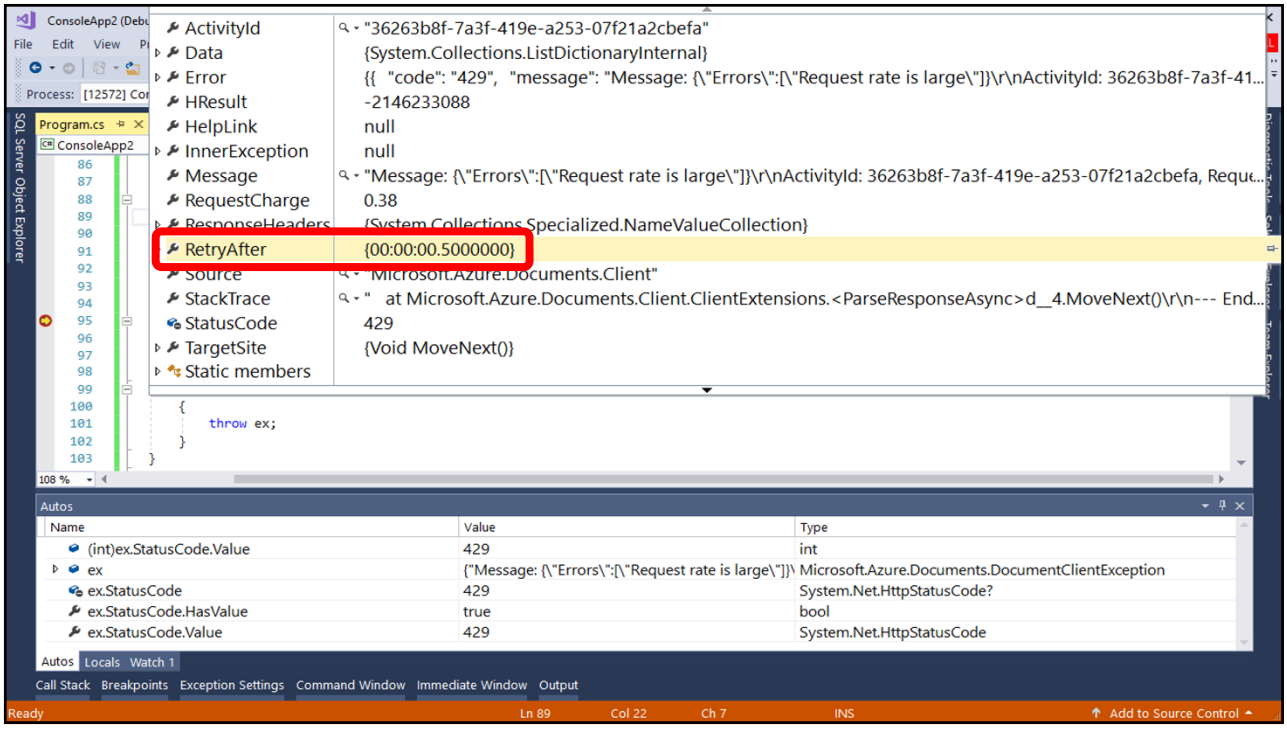

Metric - Microsoft Azure
https://portal.azure.com/#resource/subscriptions/62e2dd31-22cc-439d-aa5e-edd19758f3b5/resourcegroups/cosmos
Microsoft Azure
cosmosdbdemos - Metrics > Metric
Search re
Metric
cosmosdbdemos/hnsJAA==/hnsJAJgzCQA=
New
Dashboard
All resources
Resource groups
Recent
App Services
Virtual machines (classic)
Virtual machines
SQL databases
Cloud services (classic)
Subscriptions
Add alert
Diagnostic settings
Edit chart
Max RUs Per Second past hour
200
150
100
50
0
9:45 AM
10 AM
10:15 AM
10:30 A
MAX RUS PER SECOND
170
AVERAGE
MINIMUM
MAXIMUM
TOTAL
59.09
0
170
--

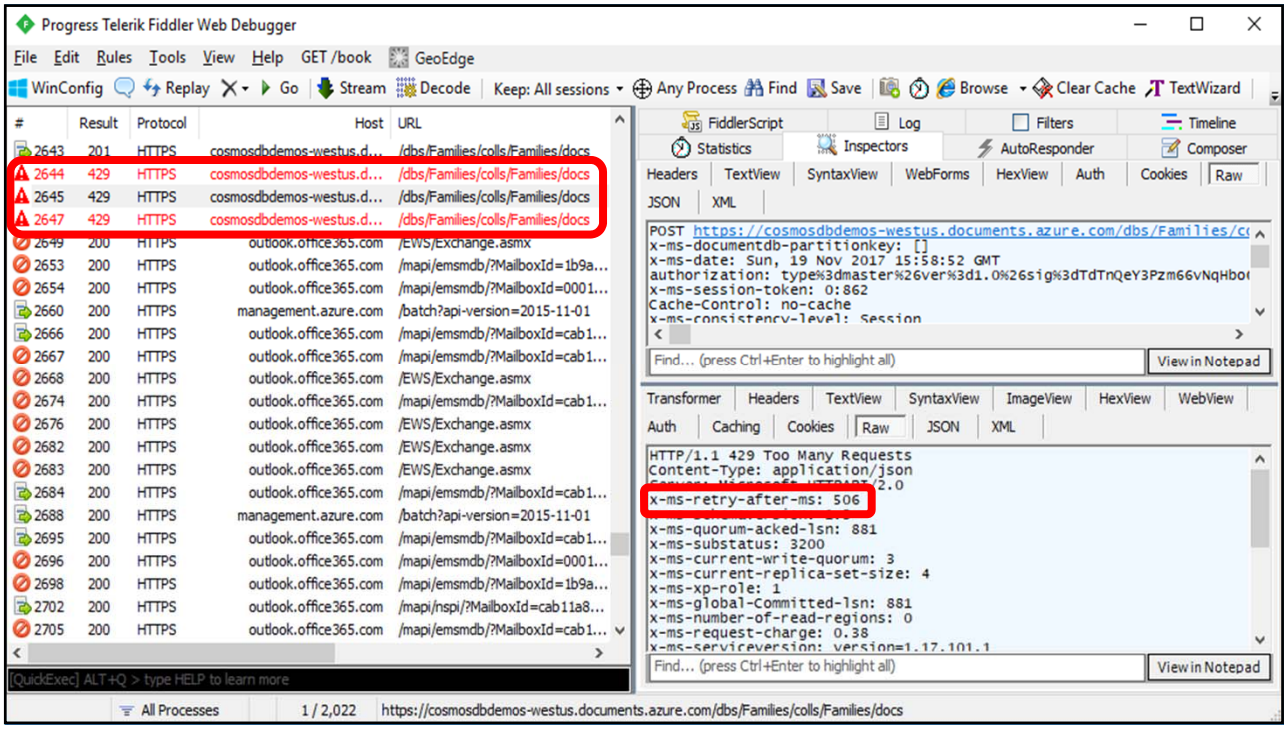

Program.cs
ConsoleApp2
ConsoleApp2.Program
RunDemo_CreateDocs()
var collection = UriFactory.CreateDocumentCollectionUri("Families", "Families");
dynamic documentDefinition = new
{
familyName = "Smith",
address = new
{
addressLine = "123 Main Street",
city = "Chicago",
state = "IL",
zipCode = "60601"
},
parents = new string[]
{
"Peter",
"Alice"
},
kids = new string[]
{
"Adam",
"Jacqueline",
"Joshua"
},
};
var result = await client.CreateDocumentAsync(collection, documentDefinition);
var consumedRUs = result.RequestCharge;
Console.WriteLine($"Cost to create document: {consumedRUs} RUs");
consumedRUs 8.95

Exceeding Reserved Throughput Limits




ConsoleApp2 (Debugging) - Microsoft Visual Studio
Program.cs
ConsoleApp2
ConsoleApp2.Program
RunDemo_CreateDocs()
try
{
var result = await client.CreateDocumentAsync(collection, documentDefinition);
Console.WriteLine("Document created successfully");
}
catch (DocumentClientException ex)
{
if (ex.StatusCode.HasValue && (int)ex.StatusCode.Value == 429)
{
Console.WriteLine($"Can't create document; request was throttled");
}
else
{
throw ex;
}
}
Autos
Name
Value
Type
(int)ex.StatusCode.Value
429
int
ex
{"Message: {\"Errors\":[\"Request ra
Microsoft.Azure.Documents.DocumentClientException
ex.StatusCode
429
System.Net.HttpStatusCode?
ex.StatusCode.HasValue
true
bool
ex.StatusCode.Value
429
System.Net.HttpStatusCode
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Ready
Ln 95
Col 25
Ch 7
INS
Add to Source Control


ActivityId
"36263b8f-7a3f-419e-a253-07f21a2cbefa"
Data
{System.Collections.ListDictionaryInternal}
Error
{{ "code": "429", "message": "Message: {\"Errors\":[\"Request rate is large\"]}\r\nActivityId: 36263b8f-7a3f-41...
HResult
-2146233088
HelpLink
null
InnerException
null
Message
"Message: {\"Errors\":[\"Request rate is large\"]}\r\nActivityId: 36263b8f-7a3f-419e-a253-07f21a2cbefa, Requ...
RequestCharge
0.38
ResponseHeaders
{System.Collections.Specialized.NameValueCollection}
RetryAfter
{00:00:00.5000000}
Source
"Microsoft.Azure.Documents.Client"
StackTrace
" at Microsoft.Azure.Documents.Client.ClientExtensions.<ParseResponseAsync>d__4.MoveNext()\r\n--- End...
StatusCode
429
TargetSite
{Void MoveNext()}
Static members
Autos
Name
Value
Type
(int)ex.StatusCode.Value
429
int
ex
{"Message: {\"Errors\":[\"Request rate is large\"]}\
Microsoft.Azure.Documents.DocumentClientException
ex.StatusCode
429
System.Net.HttpStatusCode?
ex.StatusCode.HasValue
true
bool
ex.StatusCode.Value
429
System.Net.HttpStatusCode
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Ready
Ln 89
Col 22
Ch 7
INS
Add to Source Control



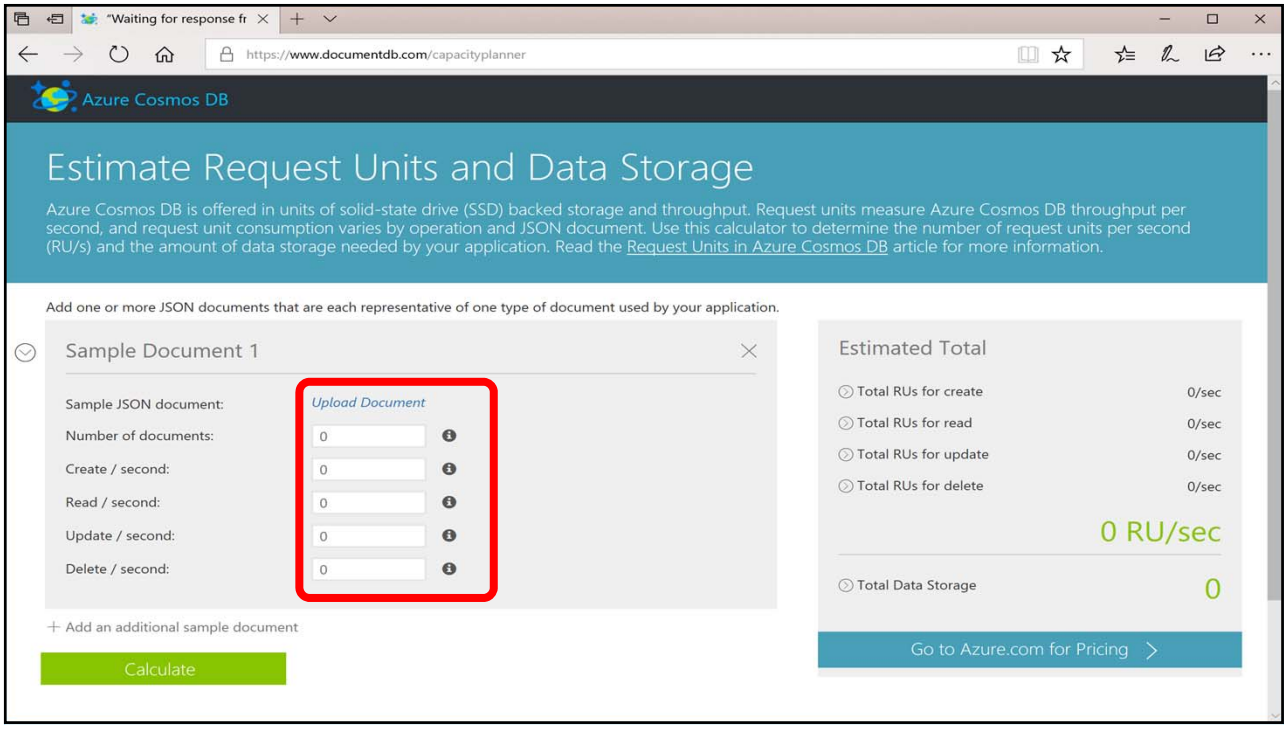
Whiteboarding the Cost

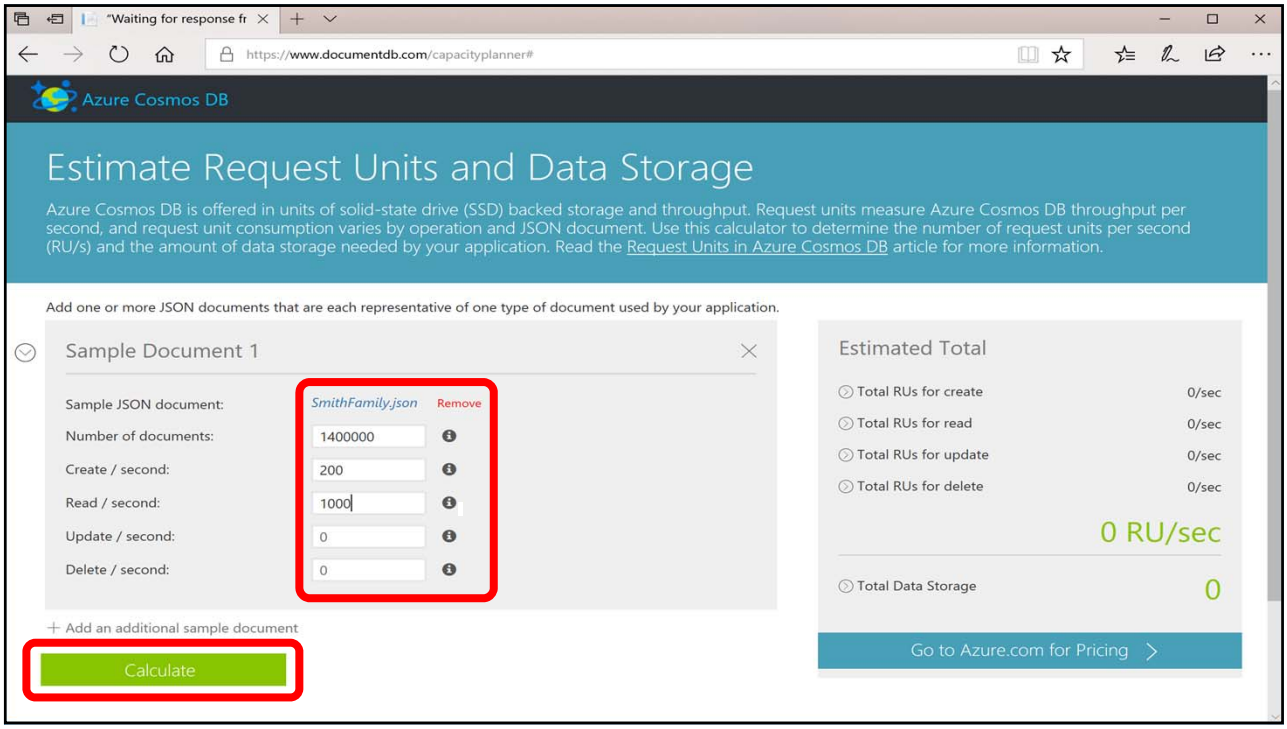
- Application checklist
  - What does a typical item look like?
  - What are the typical queries that users will run?
  - How many writes per second are required?
  - How many queries per second are required?
  - What is the acceptable consistency level?
  - What is the indexing policy?
- Estimating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

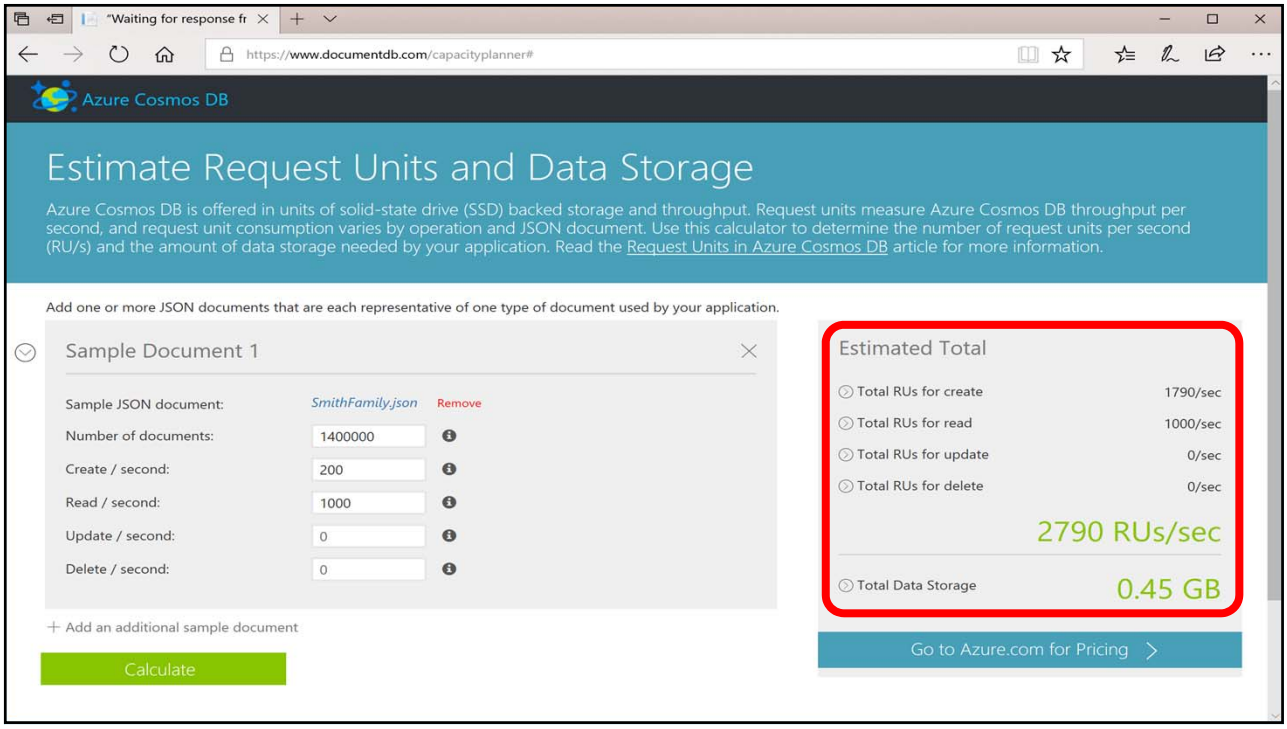
Item size	Reads/second	Writes/second	Request units
1 KB	500	100	(500 * 1) + (100 * 5) = 1,000 RU/s
1 KB	500	500	(500 * 1) + (500 * 5) = 3,000 RU/s
4 KB	500	100	(500 * 1.3) + (100 * 7) = 1,350 RU/s
4 KB	500	500	(500 * 1.3) + (500 * 7) = 4,150 RU/s
64 KB	500	100	(500 * 10) + (100 * 48) = 9,800 RU/s
64 KB	500	500	(500 * 10) + (500 * 48) = 29,000 RU/s

* Based on Session consistency indexing policy set to None.

Request Unit Calculator




"Waiting for response fr
https://www.documentdb.com/capacityplanner#
Azure Cosmos DB
Estimate Request Units and Data Storage
Azure Cosmos DB is offered in units of solid-state drive (SSD) backed storage and throughput. Request units measure Azure Cosmos DB throughput per second, and request unit consumption varies by operation and JSON document. Use this calculator to determine the number of request units per second (RU/s) and the amount of data storage needed by your application. Read the Request Units in Azure Cosmos DB article for more information.
Add one or more JSON documents that are each representative of one type of document used by your application.
Sample Document 1
Sample JSON document:
SmithFamily.json
Remove
Number of documents:
1400000
Create / second:
200
Read / second:
1000
Update / second:
0
Delete / second:
0
Add an additional sample document
Calculate
Estimated Total
Total RUs for create
0/sec
Total RUs for read
0/sec
Total RUs for update
0/sec
Total RUs for delete
0/sec
0 RU/sec
Total Data Storage
0
Go to Azure.com for Pricing


"Waiting for response fr
https://www.documentdb.com/capacityplanner#
Azure Cosmos DB
Estimate Request Units and Data Storage
Azure Cosmos DB is offered in units of solid-state drive (SSD) backed storage and throughput. Request units measure Azure Cosmos DB throughput per second, and request unit consumption varies by operation and JSON document. Use this calculator to determine the number of request units per second (RU/s) and the amount of data storage needed by your application. Read the Request Units in Azure Cosmos DB article for more information.
Add one or more JSON documents that are each representative of one type of document used by your application.
Sample Document 1
Sample JSON document:
SmithFamily.json
Remove
Number of documents:
1400000
Create / second:
200
Read / second:
1000
Update / second:
0
Delete / second:
0
Add an additional sample document
Calculate
Estimated Total
Total RUs for create
1790/sec
Total RUs for read
1000/sec
Total RUs for update
0/sec
Total RUs for delete
0/sec
2790 RUs/sec
Total Data Storage
0.45 GB
Go to Azure.com for Pricing

Pricing

SSD Storage

	A	B	C
1	1 GB	10 GB	
2	$ 0.25	$ 2.50	/month

Throughput

	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

Horizontal Partitioning

Achieving Elastic Scale

What is Partitioning?

Massive scale-out within a container

Unlimited Containers

Logical resource
composed of multiple partitions

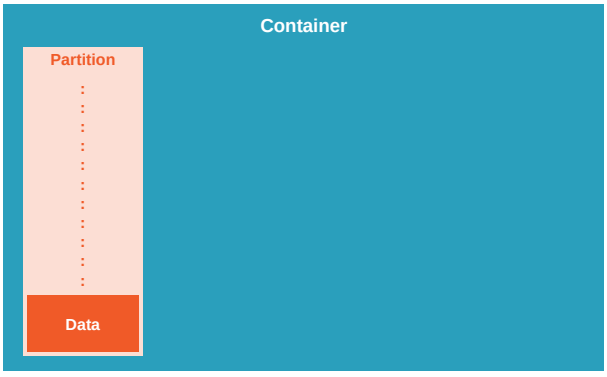
Partitions

Physical fixed-capacity data buckets

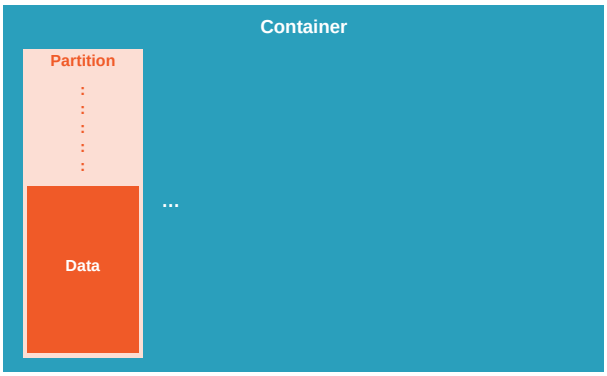
Automated Scale-Out

Cosmos DB transparently splits
partitions to manage growth

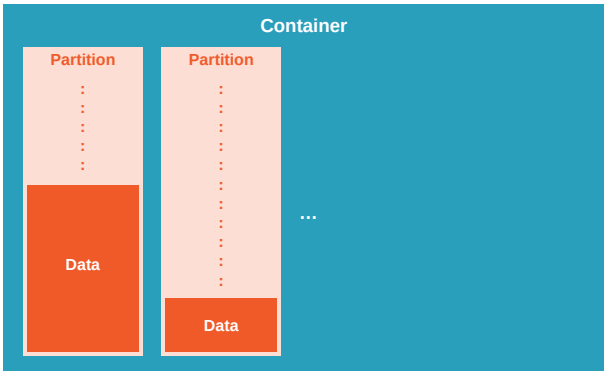
Achieving Elastic Scale



Achieving Elastic Scale



Achieving Elastic Scale



Selecting a Partition Key

Choosing the best partition key

The right choice will deliver massive scale

Partition key values are hashed

Hashed value determines the physical partition for storing each item

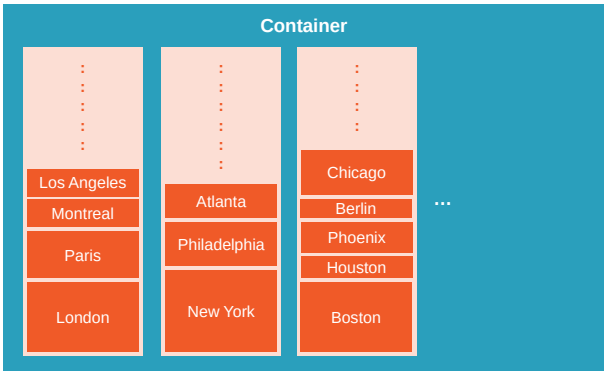
Partitions host multiple partition keys

Items with the same partition key value are physically stored together on the same partition

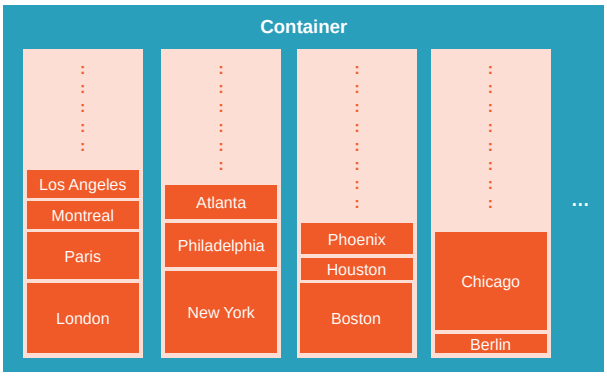
Two primary considerations

1) Boundary for query and transactions
2) No storage or performance bottlenecks

Selecting a Partition Key



Selecting a Partition Key



Choosing the Right Partition Key

Driven by data access patterns

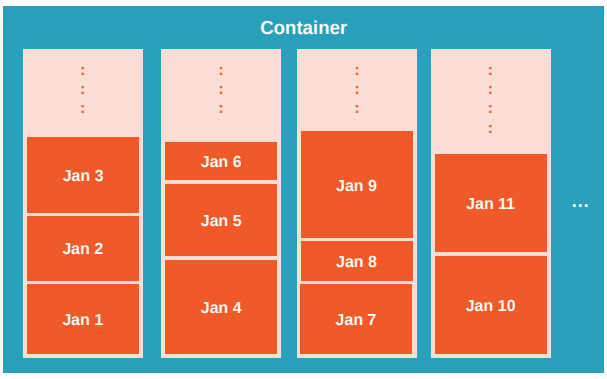
Choose a property that groups commonly queried/updated items together

User Profile Data	IoT (e.g., device state)	Multi-Tenant Architecture
User ID	Device ID	Tenant ID

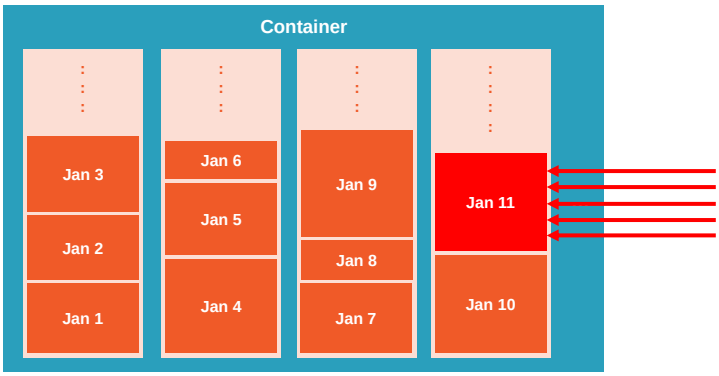
Choosing the Right Partition Key

- Generally, writes should be distributed uniformly across partitions
- For example, user profile data with a user ID and creation date
  - Partitioning by creation date
    - Bad idea! All writes of the day are directed to the same partition

Choosing the Right Partition Key



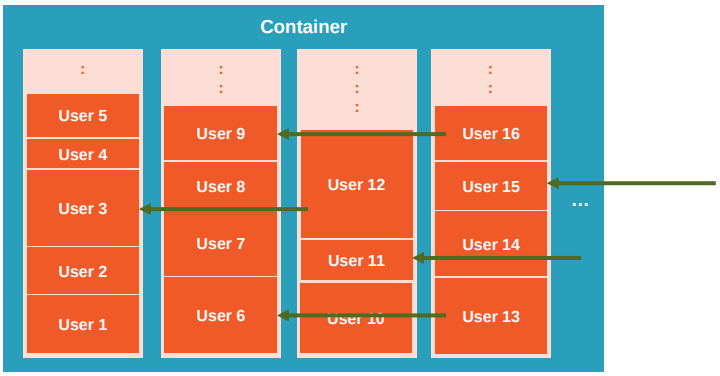
Choosing the Right Partition Key



Choosing the Right Partition Key

- Generally, writes should be distributed uniformly across partitions
- For example, user profile data with a user ID and creation date
  - Partitioning by creation date
    - Bad idea! All writes of the day are directed to the same partition
  - Partition by user ID
    - Much better! Writes are directed to different partitions per user

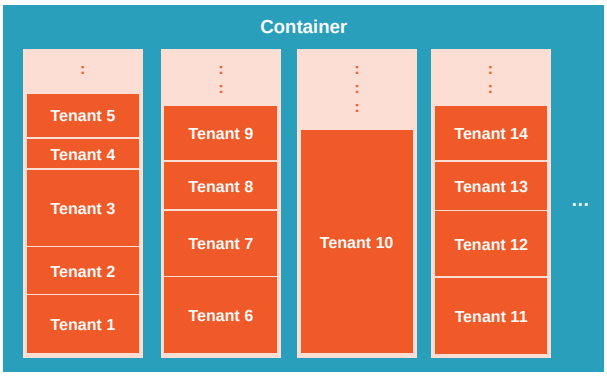
Choosing the Right Partition Key



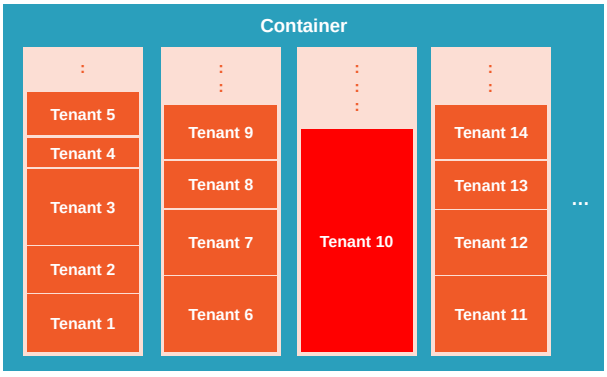
Choosing the Right Partition Key

- Generally, writes should be distributed uniformly across partitions
- For example, user profile data with a user ID and creation date
  - Partitioning by creation date
    - Bad idea! All writes of the day are directed to the same partition
  - Partition by user ID
    - Much better! Writes are directed to different partitions per user
- Create multiple containers for varying throughput needs
  - Throughput is purchased at the container level

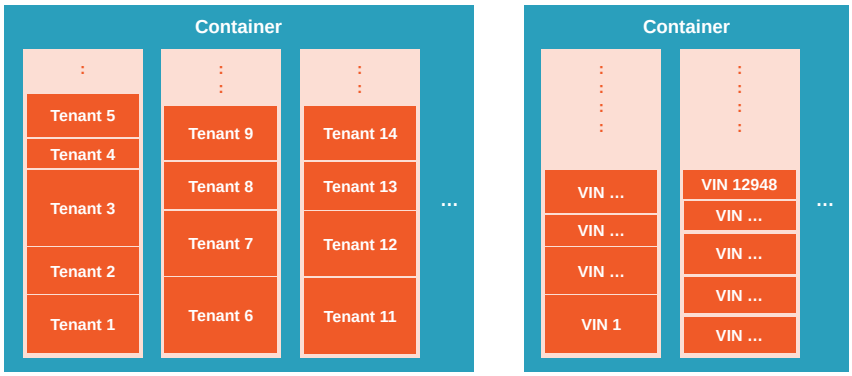
Choosing the Right Partition Key



Choosing the Right Partition Key



Choosing the Right Partition Key



Cross Partition Queries

Stored Procedures

Always scoped to a single partition key

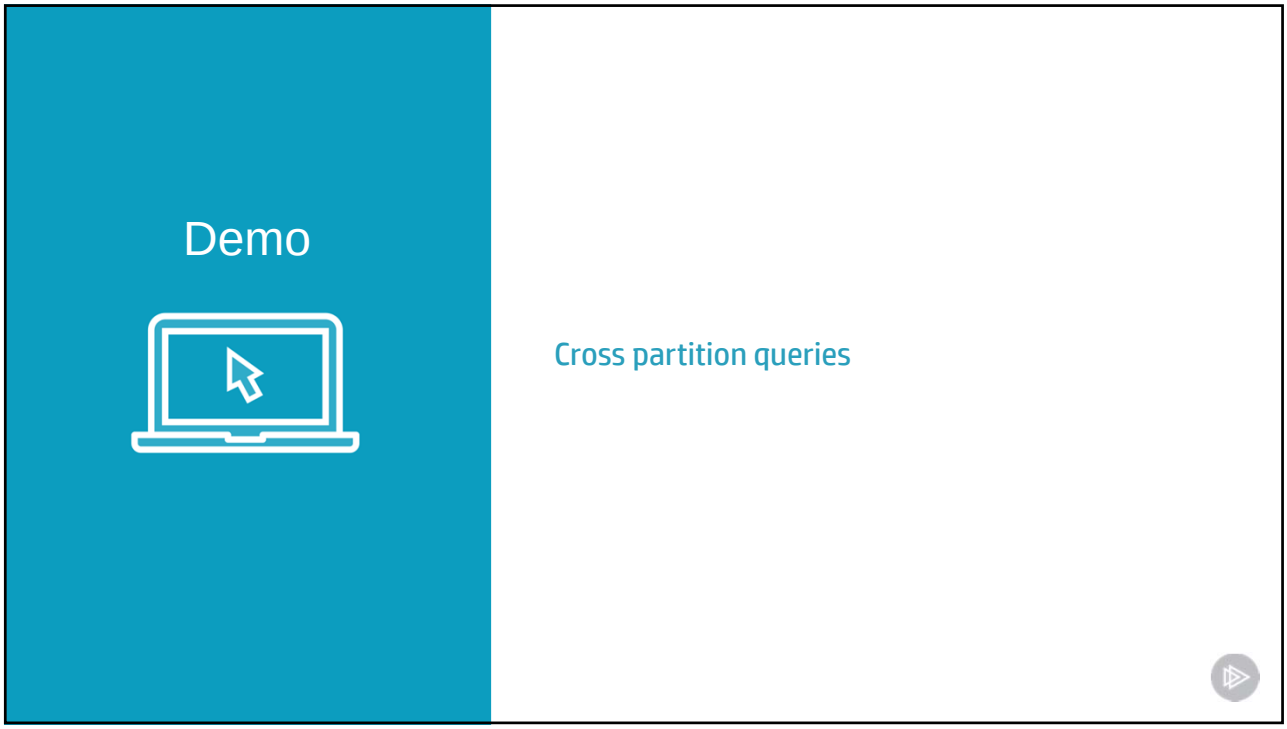
Queries

Typically scoped to a single partition key

Cross-Partition Queries

Span multiple partition keys

Fan-out execution




```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList();

    await client.DeleteDatabaseAsync(dbUri);
}
```

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList(); ≤ 7ms elapsed

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList();

    await client.DeleteDatabaseAsync(dbUri);
}
```

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'"; ≤ 261ms elapsed
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList();

    await client.DeleteDatabaseAsync(dbUri);
}
```

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);  ≤ 78ms elapsed
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList();

    await client.DeleteDatabaseAsync(dbUri);
}
```

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);  ≤ 78ms elapsed
    }
```

ex Count = 1

Data	{System.Collections.ListDictionaryInternal}
HResult	-2146233088
HelpLink	null
InnerException	{"Cross partition query is required but disabled. Please set x-ms-documentdb-query-enablecross
InnerExceptions	Count = 1
Message	"One or more errors occurred."

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message); ≤ 78ms elapsed
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList();

    await client.DeleteDatabaseAsync(dbUri);
}
```

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList(); ≤ 3ms elapsed

    await client.DeleteDatabaseAsync(dbUri);
}
```

```
using (var client = new DocumentClient(new Uri(endpoint), masterKey))
{
    var db = new Database { Id = "families" };
    await client.CreateDatabaseAsync(db);

    var coll = new DocumentCollection { Id = "families" };
    coll.PartitionKey.Paths.Add("/address/zipCode");

    await client.CreateDocumentCollectionAsync(dbUri, coll, new RequestOptions { OfferThroughput = 20000 });

    var sql = "SELECT * FROM c WHERE c.address.zipCode = '60603'";
    var query = client.CreateDocumentQuery<Document>(collUri, sql);
    var result = query.ToList();

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    query = client.CreateDocumentQuery<Document>(collUri, sql);
    try
    {
        result = query.ToList();
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
    }

    sql = "SELECT * FROM c WHERE c.address.city = 'Chicago'";
    var options = new FeedOptions { EnableCrossPartitionQuery = true, MaxDegreeOfParallelism = -1 };
    query = client.CreateDocumentQuery<Document>(collUri, sql, options);
    result = query.ToList();

    await client.DeleteDatabaseAsync(dbUri); ≤ 936ms elapsed
}
```

Global Distribution

Replication – Why?

Performance

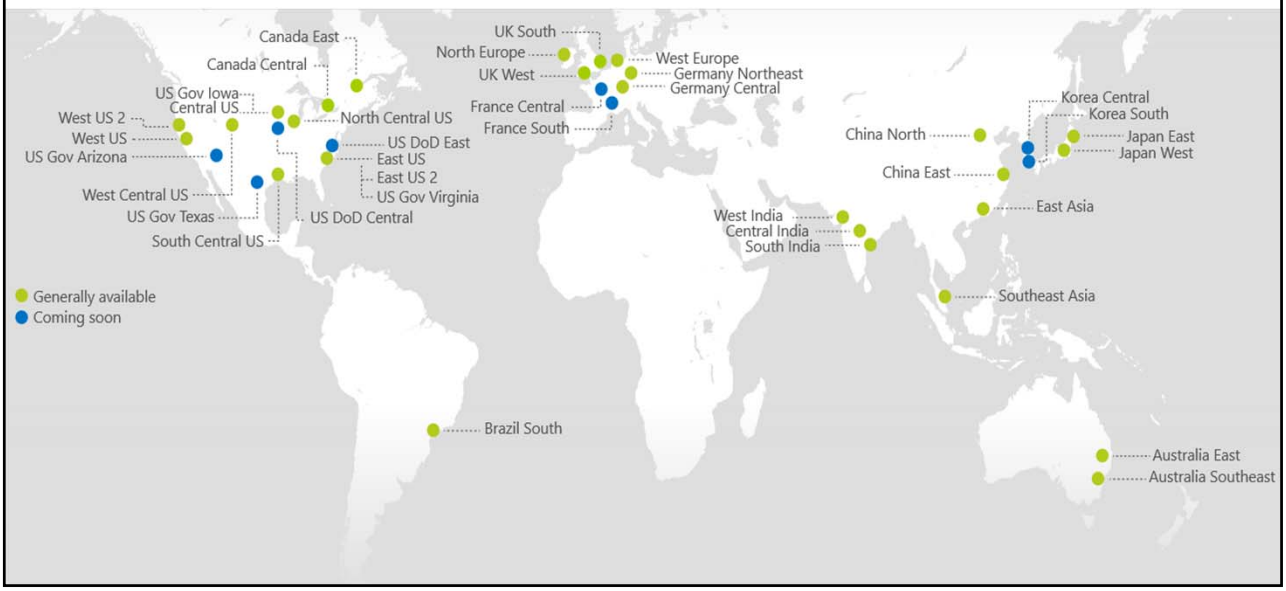
Within a region, ensures SLA on RUs purchased

Across regions, brings data closer to the consumer

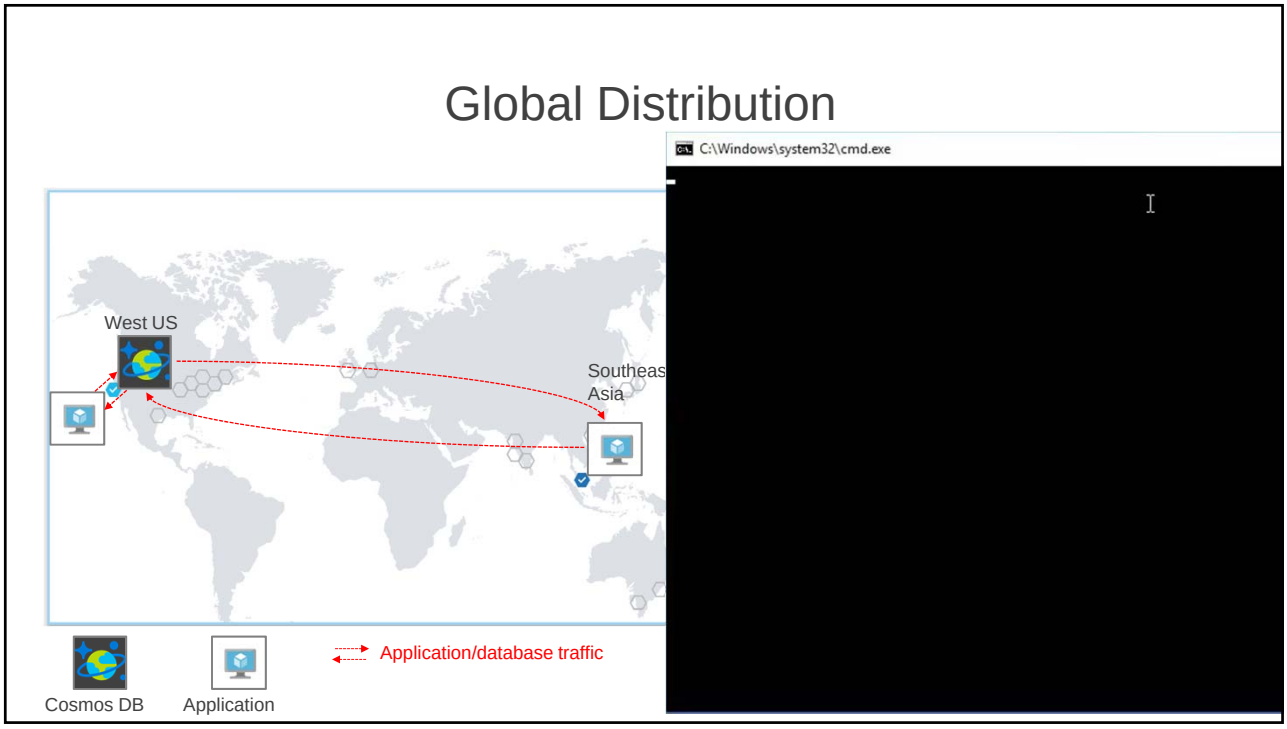
Failover

Business continuity in the event of major failure or natural disaster

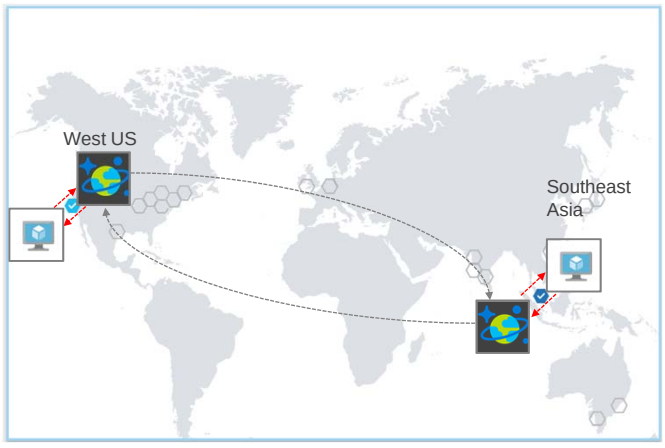
Microsoft Azure Data Centers



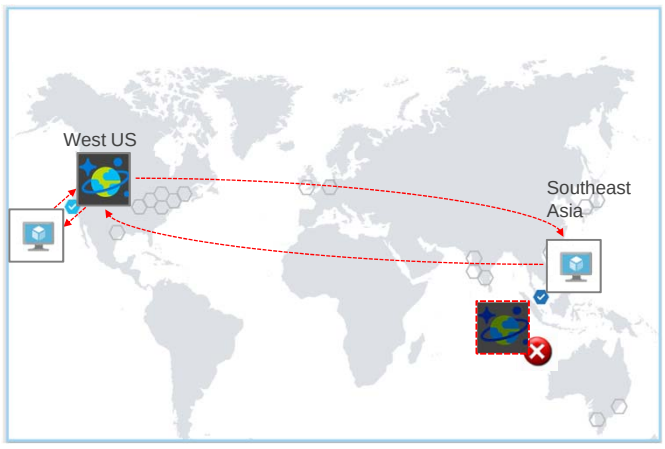

Global Distribution
C:\Windows\system32\cmd.exe
West US
Cosmos DB
Application
Application/database traffic


Global Distribution
C:\Windows\system32\cmd.exe
West US
Southeas
Asia
Cosmos DB
Application
Application/database traffic

Global Distribution

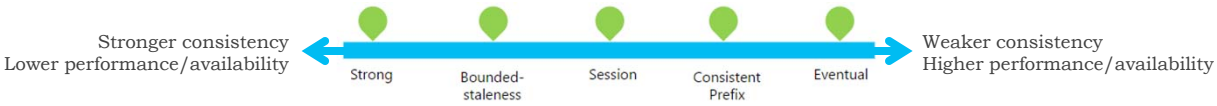


Global Distribution



Tunable Consistency

- How do you ensure consistent reads across replicas?
  - Define a consistency level



- Replication within a region
  - Data moves extremely fast (typically, within 1ms) between neighboring racks
- Global replication
  - It takes hundreds of milliseconds to move data across continents

Five Consistency Levels

Strong
No stale reads

Bounded staleness
Stale reads possible
Bounded by version, time, and region

Session
No stale reads for writers (read your own writes)
Stale reads possible for other users

Consistent prefix
Stale reads possible
Reads never see out-of-order writes

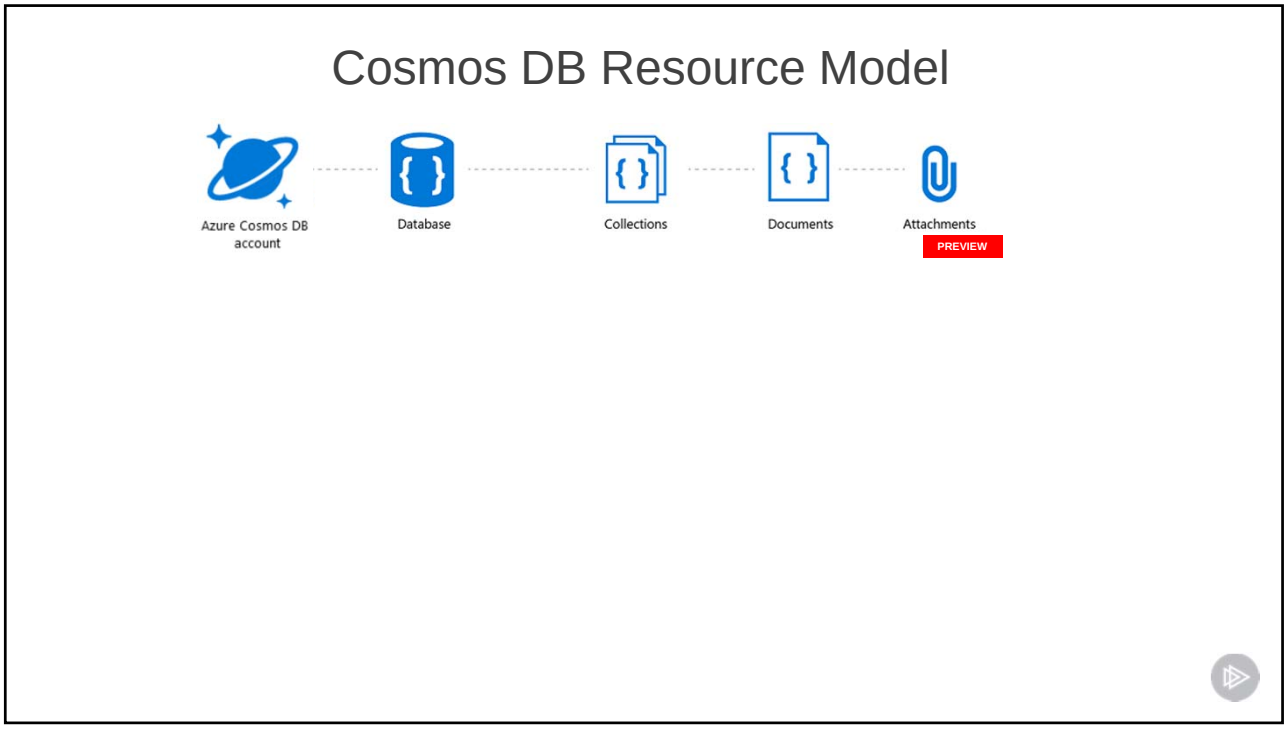
Eventual
Stale reads possible
No guaranteed order

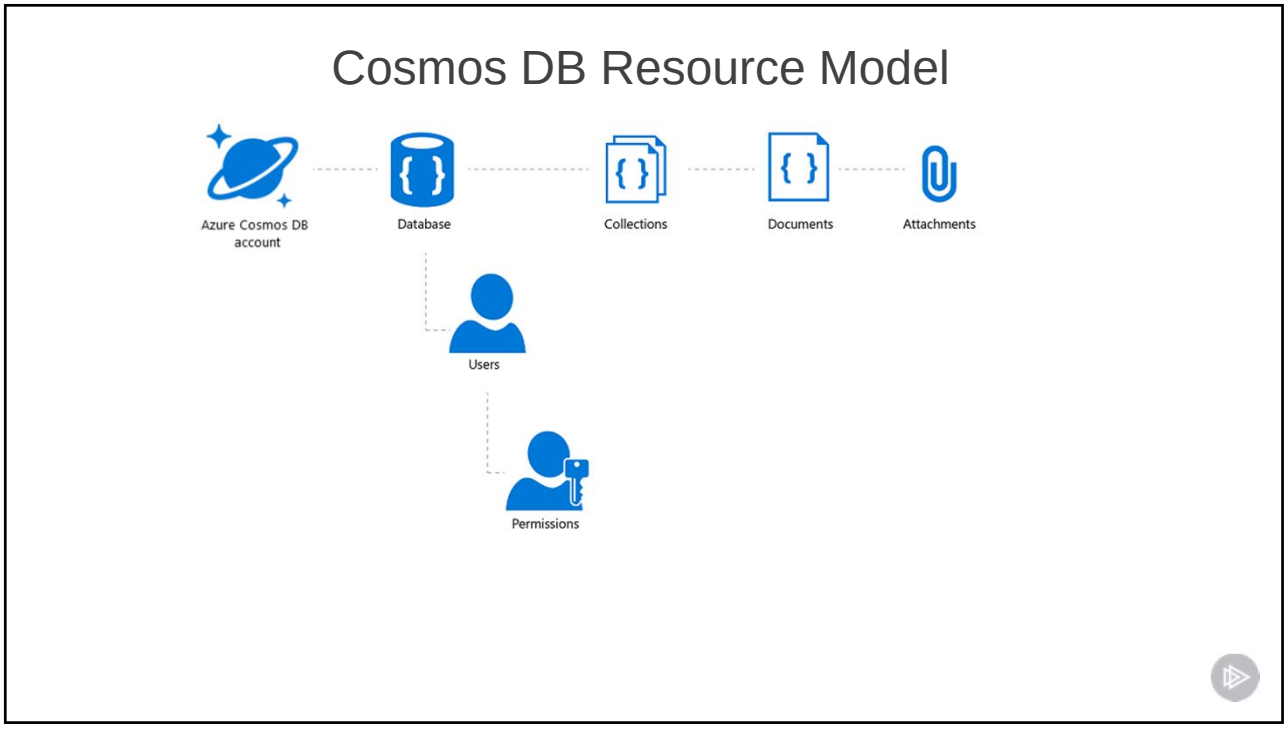
Cosmos DB Resource Model

Cosmos DB Resource Model

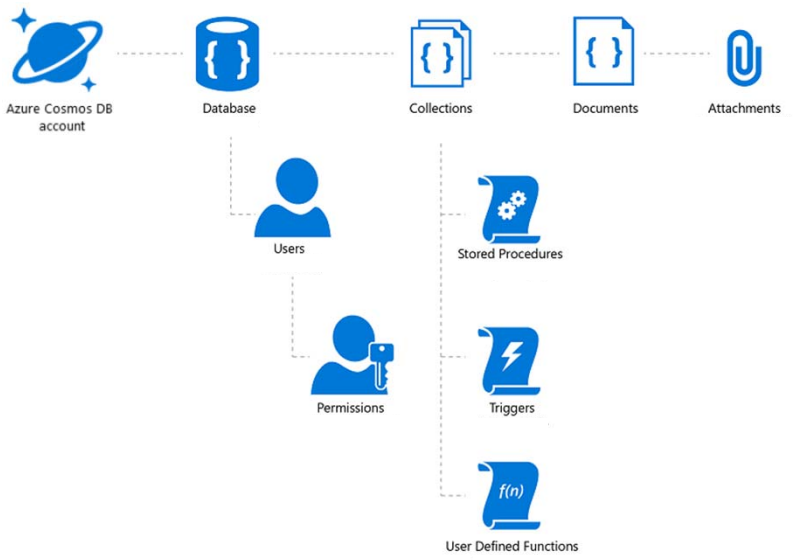
Cosmos DB Resource Model

Cosmos DB Resource Model


Cosmos DB Resource Model
Azure Cosmos DB account
Database
Collections
Documents
Attachments
PREVIEW


Cosmos DB Resource Model
Azure Cosmos DB account
Database
Collections
Documents
Attachments
Users
Permissions

Cosmos DB Resource Model



Data Migration Tool

Open source project

Microsoft Download Center
http://www.microsoft.com/en-us/download/details.aspx?id=46436

Documentation and How-To
https://azure.microsoft.com/en-us/documentation/articles/documentdb-

Source code on GitHub
https://github.com/azure/azure-documentdb-datamigrationtool

Import from

- SQL Server
- JSON
- CSV
- MongoDB
- Azure Table Storage
- and more…

Querying with the SQL API

Rich Query with SQL

Cosmos DB SQL

Special version of SQL designed for JSON
.NET SDK also includes a LINQ provider

Familiar keywords

SELECT, FROM, WHERE, JOIN,
IN, BETWEEN, ORDER BY

Work with JSON

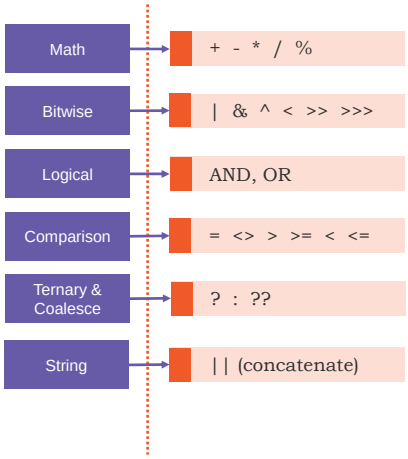
Descend into any subsection of a document
Iterate nested arrays for intra-document join
Return documents as-is, or as custom shape

Example

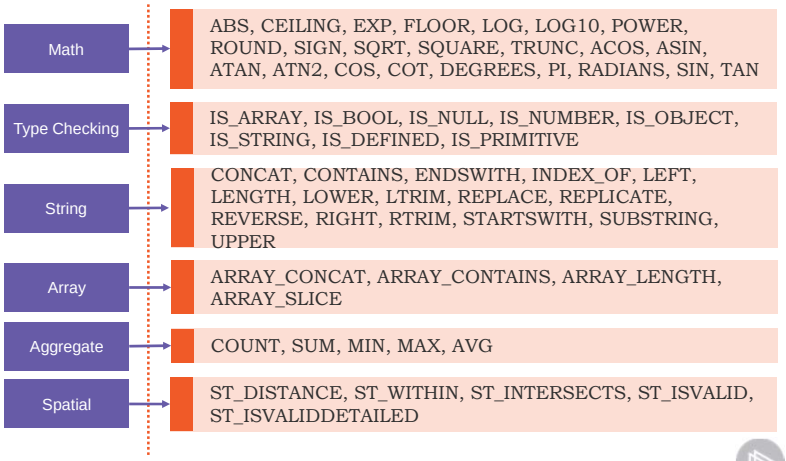
```
SELECT ch.name, ch.birthdate
FROM Families AS f
JOIN ch IN f.children
WHERE f.address.state = 'CA'
```

SQL Operators and Functions

Common operators



Built-in functions



Using the Table API

What is the Table API?

Replaces Azure Table Storage

Implement a key-value data model

Key-value data model

Key = PartitionKey + RowKey

Value = key-value pairs!

Leverage Cosmos DB back end

Predictable throughput

Global distribution

Automatic indexing

SQL API vs. Table API

Cosmos DB	SQL API	Table API

SQL API vs. Table API

Cosmos DB	SQL API	Table API
Container	Collection	Table

SQL API vs. Table API

Cosmos DB	SQL API	Table API
Container	Collection	Table
Item	Document	Row

Document (SQL API)

```
{
  "genre": "sci-fi",
  "id": "Star Trek II",
  "year": 1982,
  "length": "1h, 53m",
  "description": "Khan is back!"
}
```

Row (Table API)

PartitionKey	RowKey	Year	Length	Description
sci-fi	Star Trek II	1982	1h, 53m	Khan is back!

SQL API vs. Table API

Cosmos DB	SQL API	Table API
Container	Collection	Table
Item	Document	Row
Partition Key	Any property	PartitionKey

SQL API vs. Table API

Cosmos DB	SQL API	Table API
Container	Collection	Table
Item	Document	Row
Partition Key	Any property	PartitionKey
ID	id	RowKey

Why Use the Table API?

Migrate existing applications

Just change the connection string

Upgrade SDK

Cosmos DB Table SDK uses native protocol

No advantage for new applications

Table API is a layer over SQL API

Working with the Gremlin API

Cosmos DB Graph Database

Graph container

Horizontal partitioning, provisioned throughput, global distribution, indexing

Vertex and Edge objects

Entities and relationships
Both can hold arbitrary key-value pairs

Create/Retrieve/Update/Delete

Gremlin and GraphSON

Apache TinkerPop
http://tinkerpop.apache.org

Focus on relationships

Chain relationship queries with Gremlin

Cosmos DB is (currently) the *only* managed cloud graph database

Others are following in hot pursuit

Graph Database Scenarios

Complex relationships

Many "many-to-many" relationships

Excessive JOINs

Analyze interconnected data and relationships

Typical graph applications

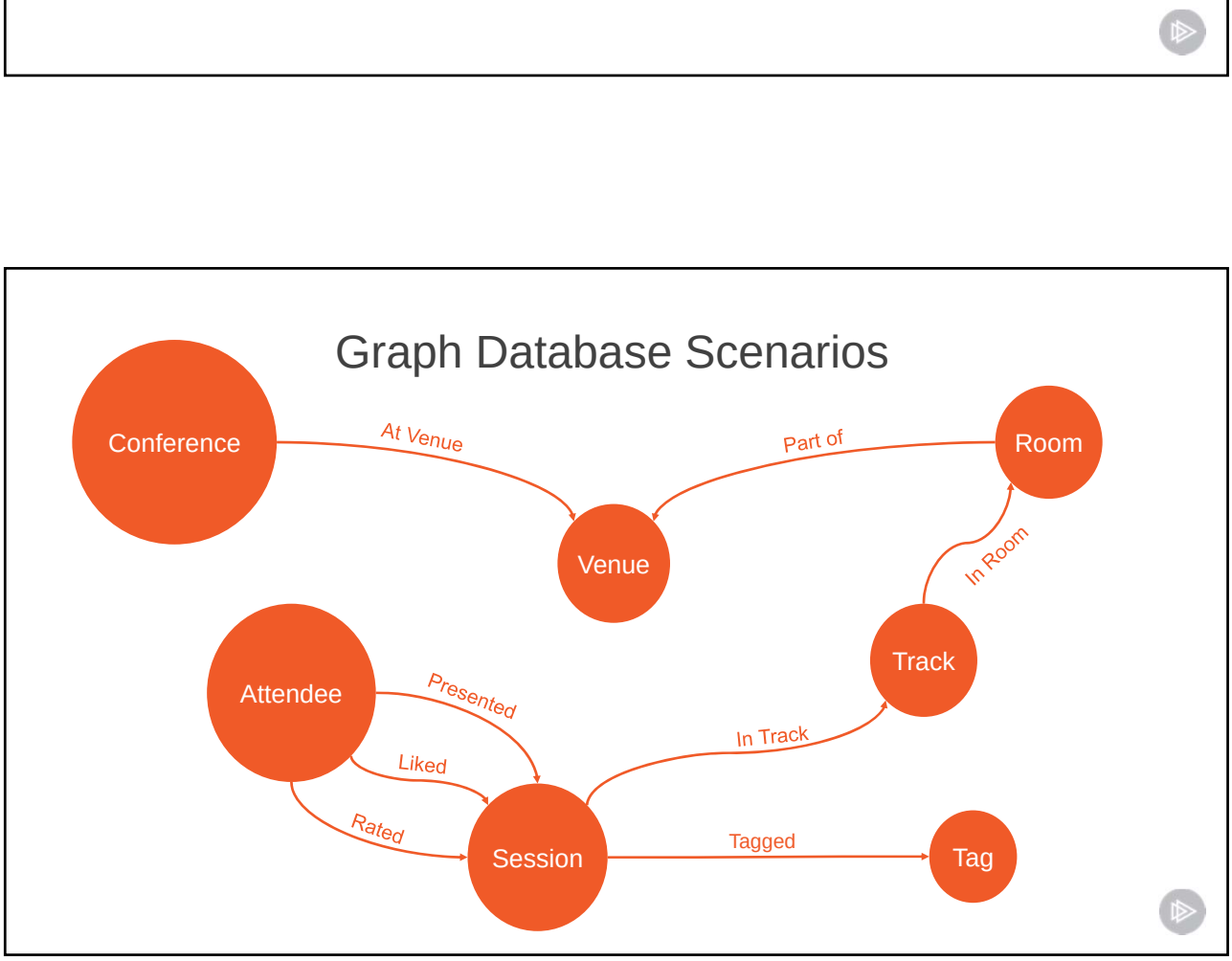
Social networks

Recommendation engines

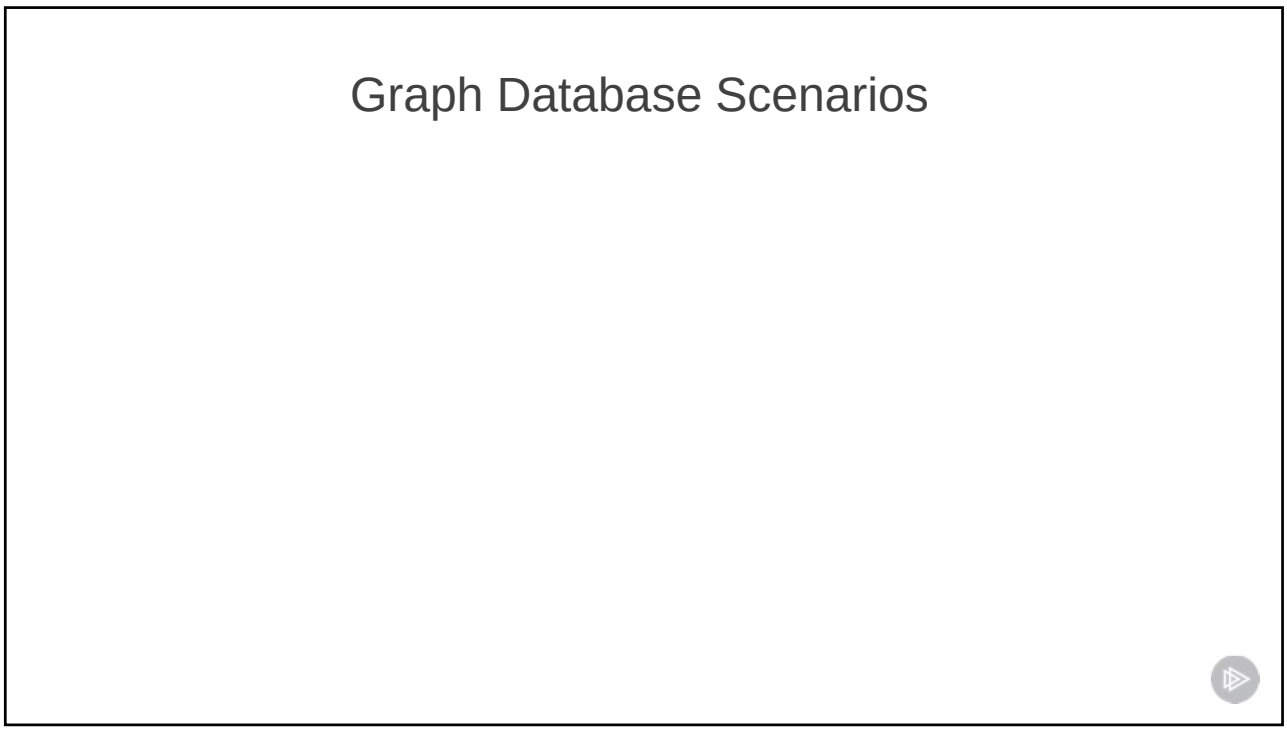
Knowledge graphs

Many more…

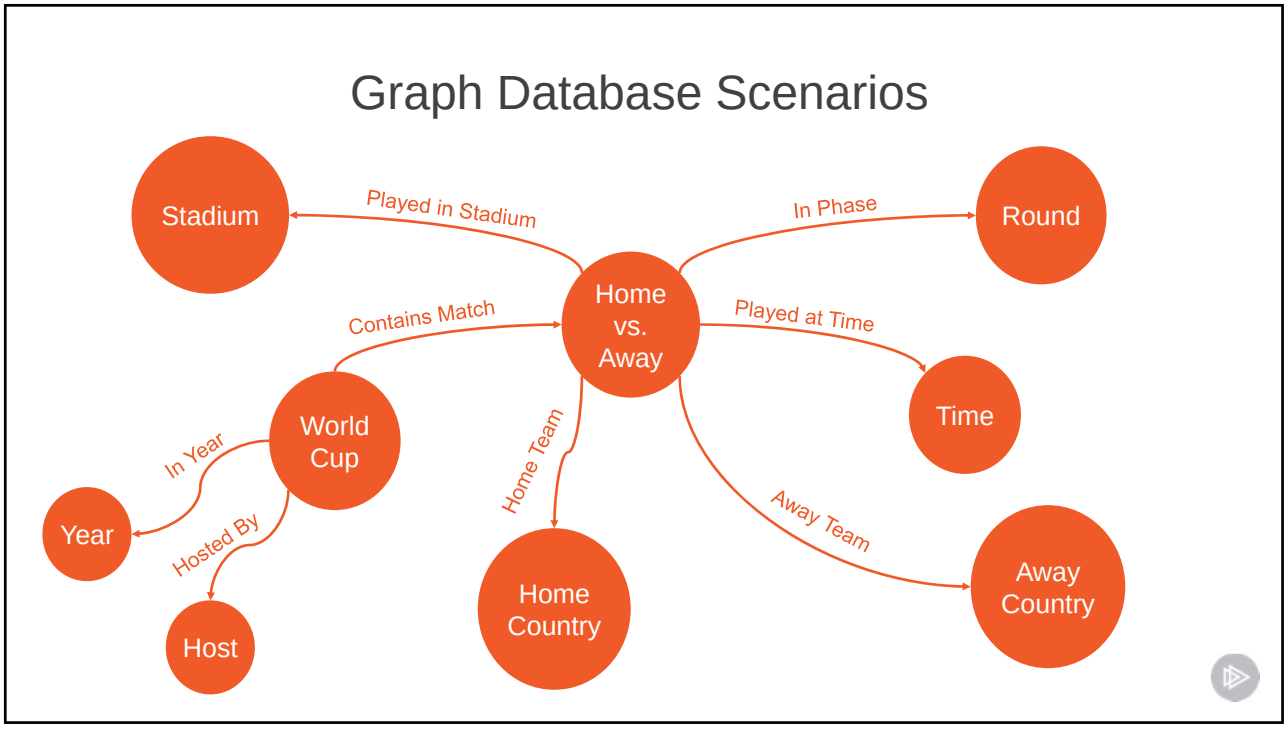
Graph Database Scenarios



Graph Database Scenarios



Graph Database Scenarios



GraphSON and Gremlin

GraphSON

Gremlin JSON wire format

ID, label

Arbitrary key-value pairs

Two types of GraphSON objects

Vertex = Entity

Edge = Relationship

Gremlin .NET

Issue Gremlin queries from C# applications

Create and query vertices and edges

Vertices and Edges

Vertex and Edge properties

id (within partition key)

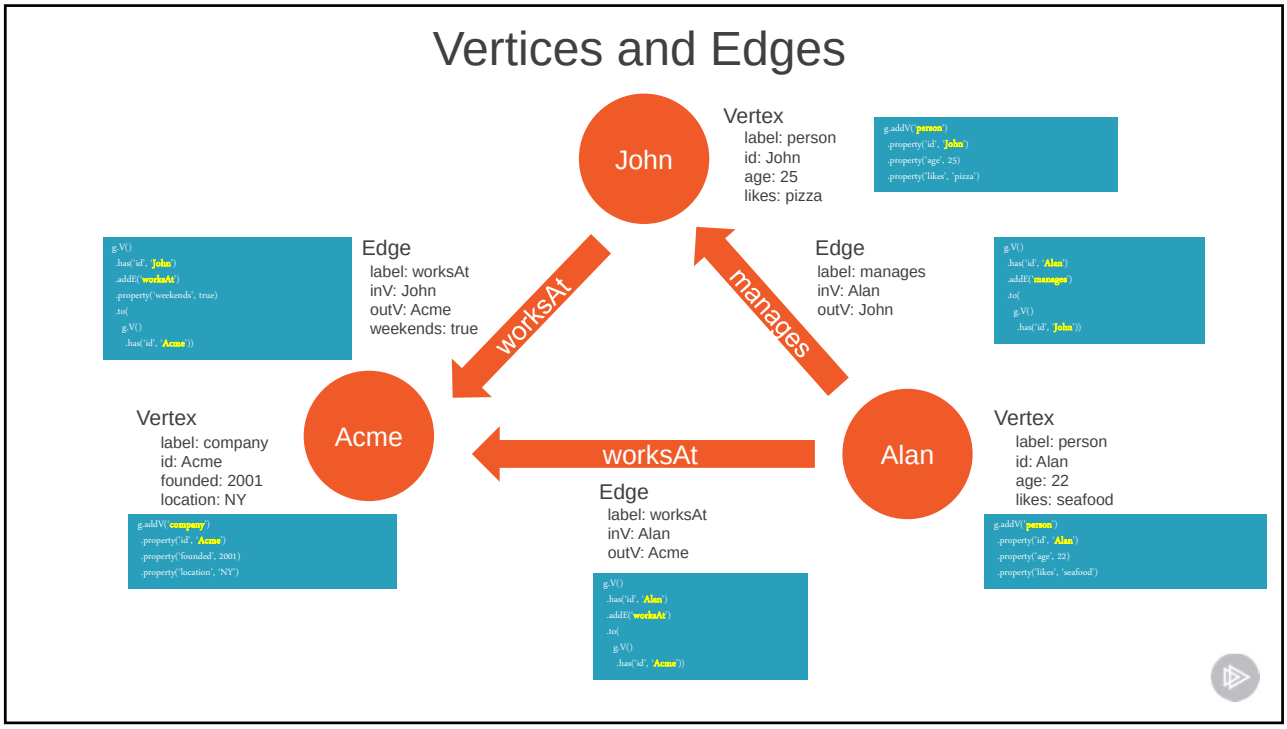
label (type)

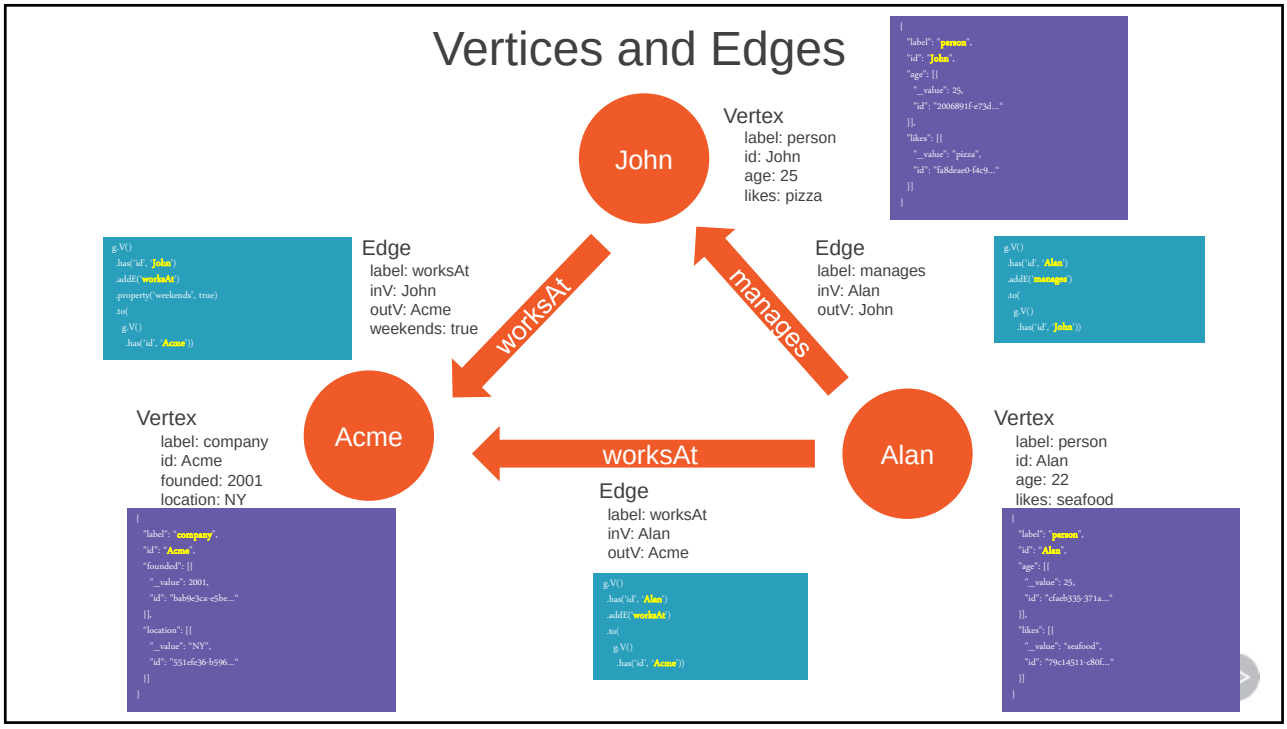
Additional arbitrary properties (including the partition key)

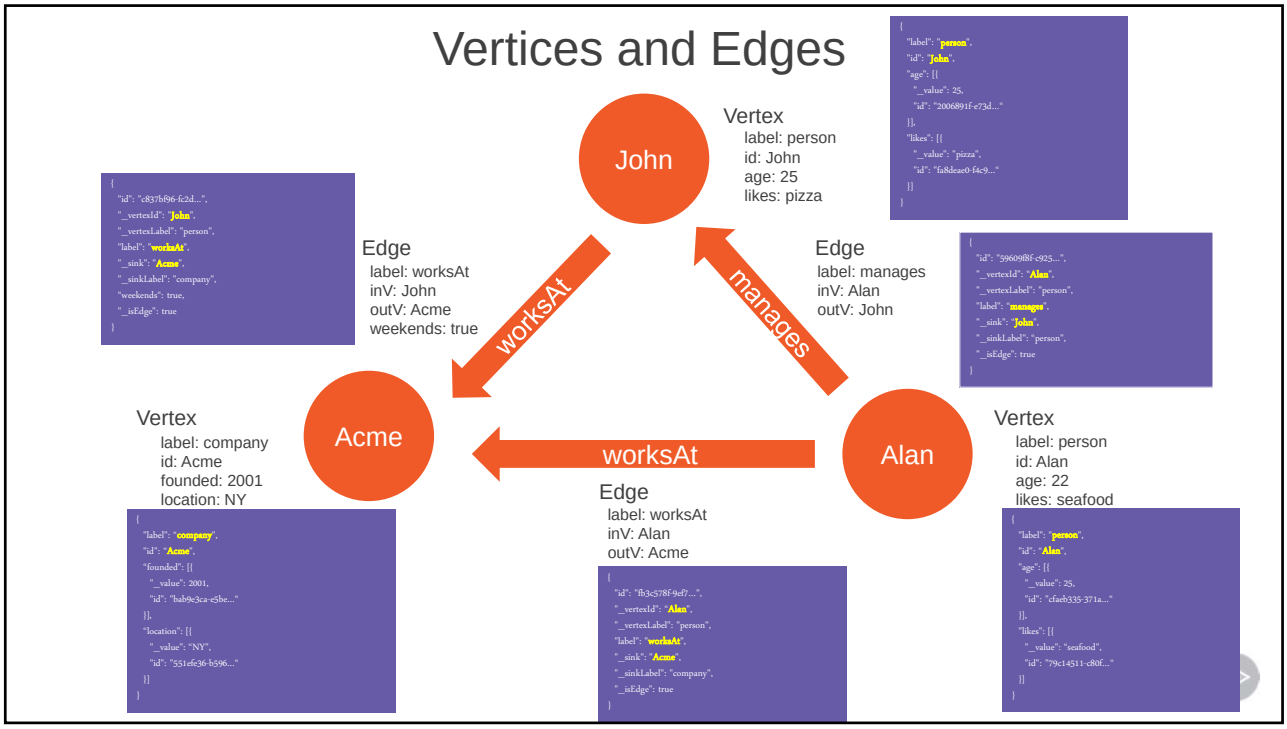
Additional Edge properties

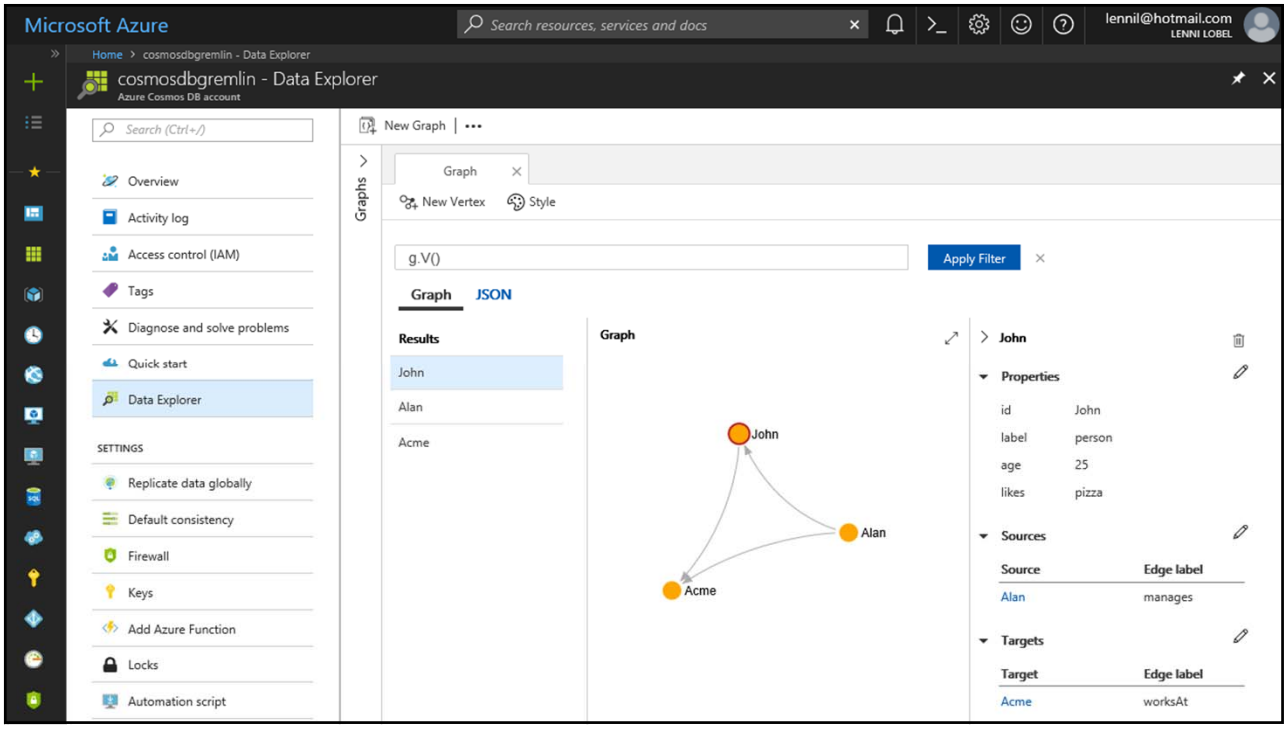
Cardinality (in-and-out vertices)

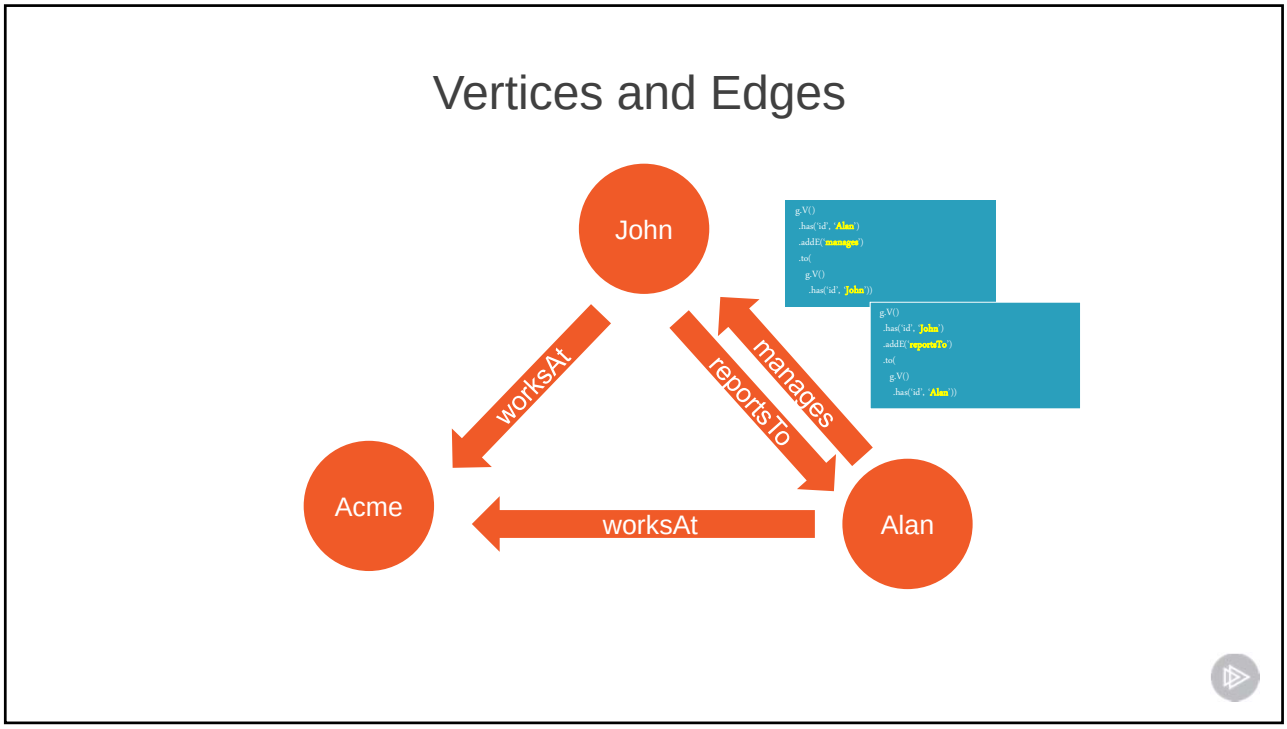
Create two edges for bi-directionality

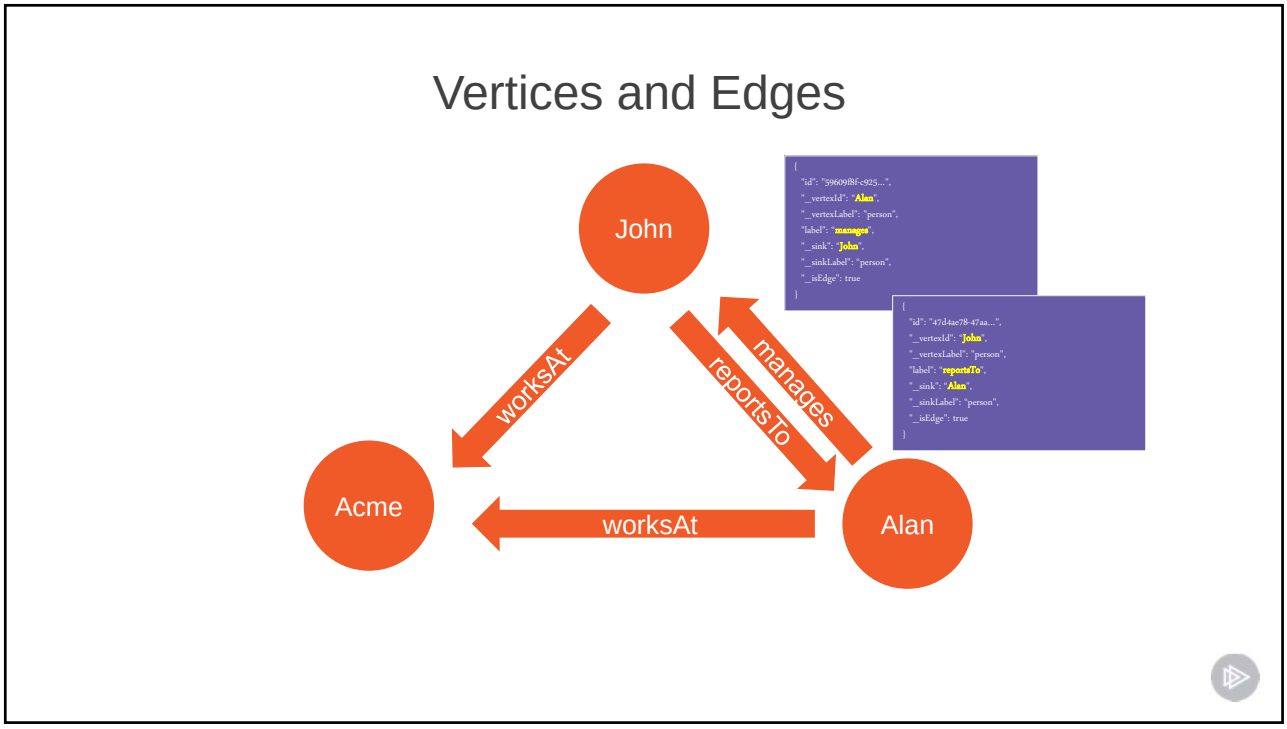

Vertices and Edges
John
Vertex
label: person
id: John
age: 25
likes: pizza
g.addV('person')
.property('id', 'John')
.property('age', 25)
.property('likes', 'pizza')
g.V()
.has('id', 'John')
.addE('worksAt')
.property('weekends', true)
.to(
g.V()
.has('id', 'Acme'))
Edge
label: worksAt
inV: John
outV: Acme
weekends: true
worksAt
manages
Edge
label: manages
inV: Alan
outV: John
g.V()
.has('id', 'Alan')
.addE('manages')
.to(
g.V()
.has('id', 'John'))
Acme
Alan
Vertex
label: company
id: Acme
founded: 2001
location: NY
g.addV('company')
.property('id', 'Acme')
.property('founded', 2001)
.property('location', 'NY')
worksAt
Edge
label: worksAt
inV: Alan
outV: Acme
g.V()
.has('id', 'Alan')
.addE('worksAt')
.to(
g.V()
.has('id', 'Acme'))
Vertex
label: person
id: Alan
age: 22
likes: seafood
g.addV('person')
.property('id', 'Alan')
.property('age', 22)
.property('likes', 'seafood')


Vertices and Edges
John
Vertex
label: person
id: John
age: 25
likes: pizza
{
"label": "person",
"id": "John",
"age": [{
"_value": 25,
"id": "2006891f-e73d..."
}],
"likes": [{
"_value": "pizza",
"id": "fa8deae0-f4c9..."
}]
}
g.V()
.has('id', 'John')
.addE('worksAt')
.property('weekends', true)
.to(
g.V()
.has('id', 'Acme'))
Edge
label: worksAt
inV: John
outV: Acme
weekends: true
worksAt
manages
Edge
label: manages
inV: Alan
outV: John
g.V()
.has('id', 'Alan')
.addE('manages')
.to(
g.V()
.has('id', 'John'))
Acme
Alan
Vertex
label: company
id: Acme
founded: 2001
location: NY
{
"label": "company",
"id": "Acme",
"founded": [{
"_value": 2001,
"id": "bab9e3ca-e5be..."
}],
"location": [{
"_value": "NY",
"id": "551e6e36-b596..."
}]
}
worksAt
Edge
label: worksAt
inV: Alan
outV: Acme
g.V()
.has('id', 'Alan')
.addE('worksAt')
.to(
g.V()
.has('id', 'Acme'))
Vertex
label: person
id: Alan
age: 22
likes: seafood
{
"label": "person",
"id": "Alan",
"age": [{
"_value": 25,
"id": "cfaeb335-371a..."
}],
"likes": [{
"_value": "seafood",
"id": "79c14511-c80f..."
}]
}


Vertices and Edges
John
Vertex
label: person
id: John
age: 25
likes: pizza
Edge
label: worksAt
inV: John
outV: Acme
weekends: true
worksAt
manages
Edge
label: manages
inV: Alan
outV: John
Acme
Vertex
label: company
id: Acme
founded: 2001
location: NY
worksAt
Alan
Edge
label: worksAt
inV: Alan
outV: Acme
Vertex
label: person
id: Alan
age: 22
likes: seafood


Microsoft Azure
Search resources, services and docs
lennil@hotmail.com
LENNI LOBEL
Home > cosmosdbgremlin - Data Explorer
cosmosdbgremlin - Data Explorer
Azure Cosmos DB account
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Quick start
Data Explorer
SETTINGS
Replicate data globally
Default consistency
Firewall
Keys
Add Azure Function
Locks
Automation script
New Graph
Graphs
Graph
New Vertex
Style
g.V()
Apply Filter
Graph
JSON
Results
John
Alan
Acme
Graph
John
Alan
Acme
John
Properties
id John
label person
age 25
likes pizza
Sources
Source Edge label
Alan manages
Targets
Target Edge label
Acme worksAt


Vertices and Edges
John
Acme
Alan
worksAt
reportsTo
manages
worksAt
g.V()
.has('id', 'Alen')
.addE('manages')
.to(
g.V()
.has('id', 'John'))
g.V()
.has('id', 'John')
.addE('reportsTo')
.to(
g.V()
.has('id', 'Alen'))


Vertices and Edges
John
Acme
Alan
worksAt
reportsTo
manages
worksAt
{
"id": "59609f8f-c925...",
"_vertexId": "Alen",
"_vertexLabel": "person",
"label": "manages",
"_sink": "John",
"_sinkLabel": "person",
"_isEdge": true
}
{
"id": "47d4ae78-47aa...",
"_vertexId": "John",
"_vertexLabel": "person",
"label": "reportsTo",
"_sink": "Alen",
"_sinkLabel": "person",
"_isEdge": true
}

Writing Gremlin Queries

Functional language

Chain multiple steps together

Define Vertices and Edges

.addV('label')

.addE('label')

.property('key', 'value')

Query on filters and relationships

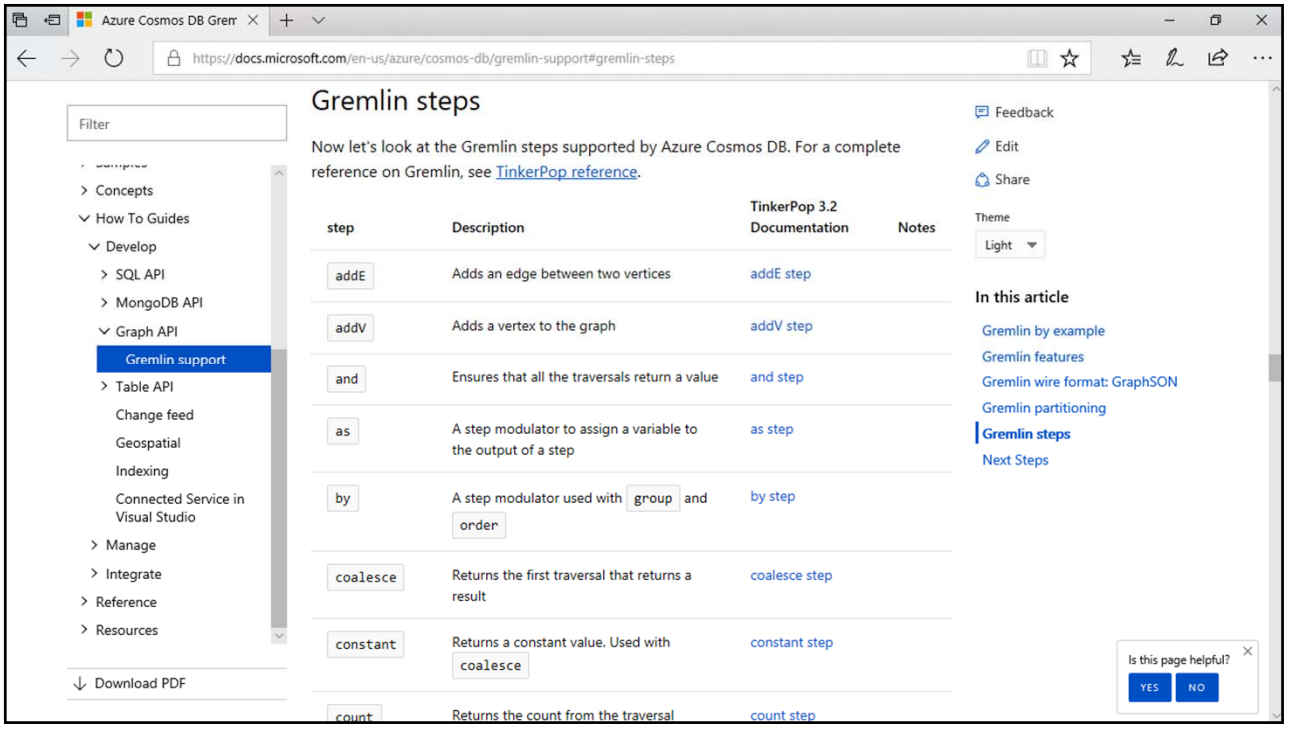
.V('label')

.out('label')

.has('property', condition)

Cosmos DB Gremlin support

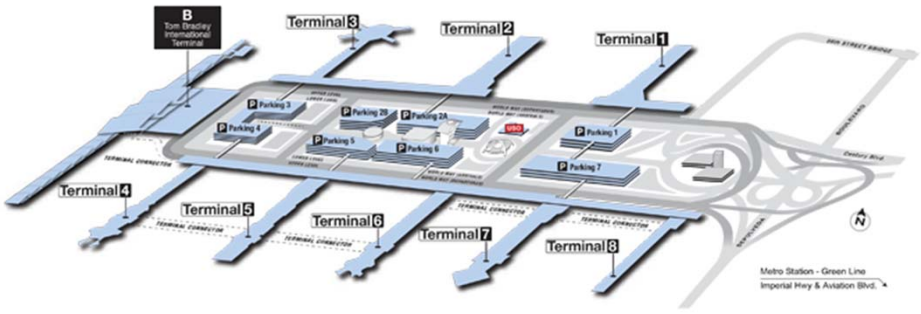
https://aka.ms/gremlin-support

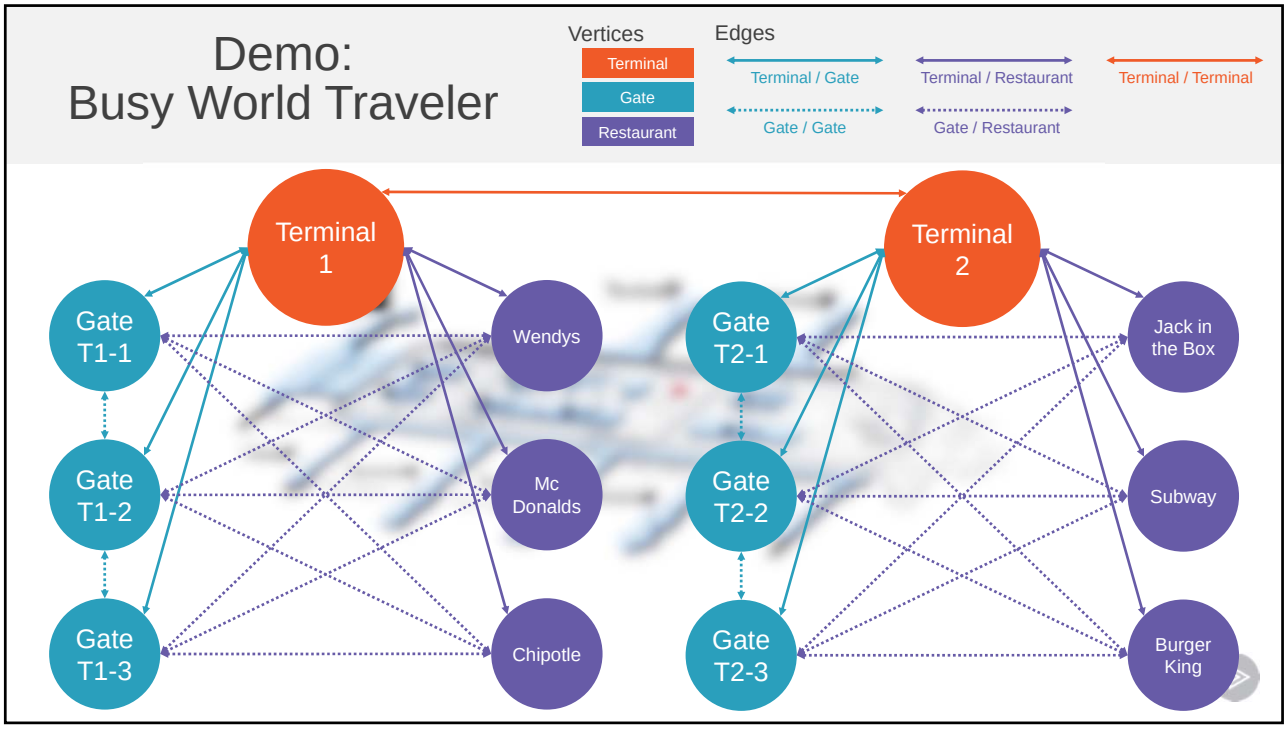


Demo

Busy world traveler

Demo:
Busy World Traveler





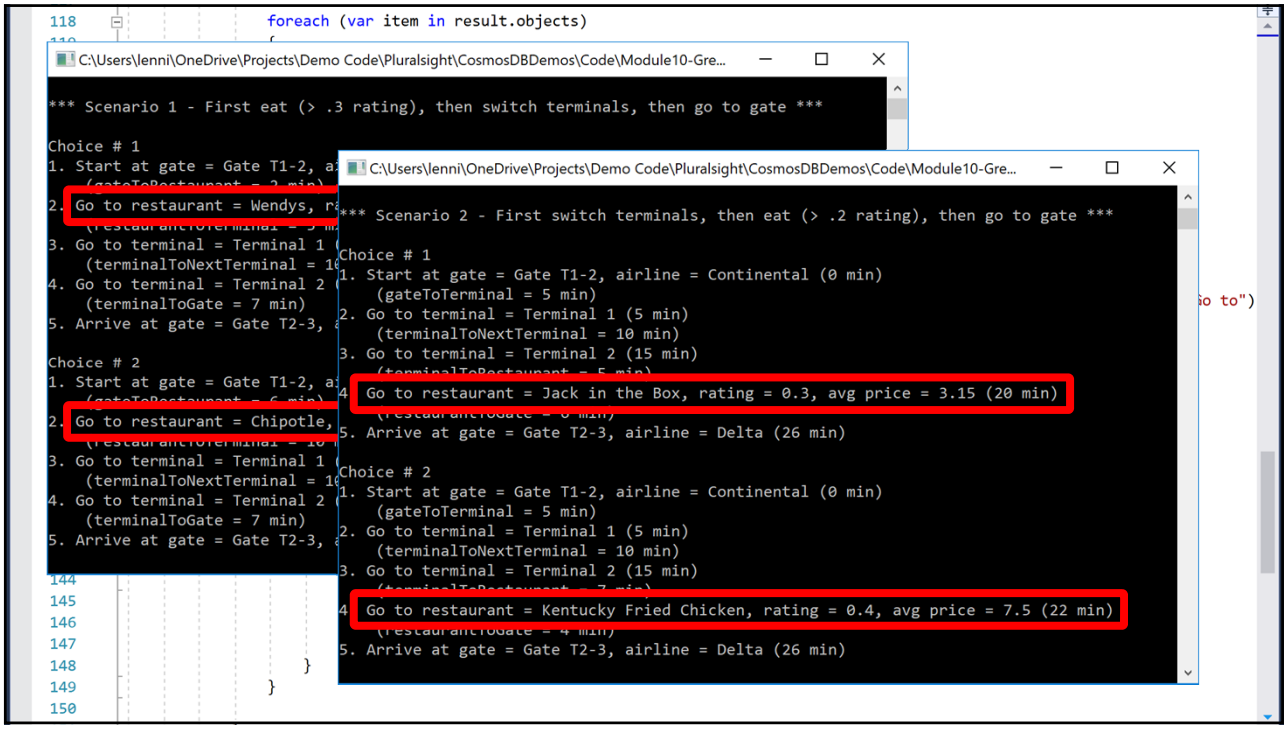

```
var firstEatThenSwitchTerminals = @"
    // Start at T1, Gate 2
        g.V('Gate T1-2')

    // Traverse edge from gate to restaurants
        .outE('gateToRestaurant')
        .inV()

    // Filter for restaurants with a rating of .3 or higher
        .has('rating', gt(0.3))

    // Traverse edge from restaurant back to terminal (T1)
        .outE('restaurantToTerminal')
        .inV()

    // Traverse edge from terminal to next terminal (T2)
        .outE('terminalToNextTerminal')
        .inV()

    // Traverse edge from terminal (T2) to gates
        .outE('terminalToGate')
        .inV()

    // Filter for destination gate T2, Gate 3
        .has('id', 'Gate T2-3')

    // Show the possible paths
        .path()
";
```

```
var firstSwitchTerminalsThenEat = @"
    // Start at T1, Gate 2
        g.V('Gate T1-2')

    // Traverse edge from gate to terminal T1
        .outE('gateToTerminal')
        .inV()

    // Traverse edge from terminal to next terminal (T2)
        .outE('terminalToNextTerminal')
        .inV()

    // Traverse edge from terminal to restaurants
        .outE('terminalToRestaurant')
        .inV()

    // Filter for restaurants with a rating of .2 or higher
        .has('rating', gt(0.2))

    // Traverse edge from restaurant back to gates
        .outE('restaurantToGate')
        .inV()

    // Filter for destination gate T2, Gate 3
        .has('id', 'Gate T2-3')

    // Show the possible paths
        .path()
";
```

```
private static async Task RunAirportQuery(GremlinClient client, string gremlinCode)
{
    var results = await client.SubmitAsync<dynamic>(gremlinCode);

    var count = 0;

    foreach (var resultDict in results)
    {
        count++;
        dynamic result = JsonConvert.DeserializeObject(JsonConvert.SerializeObject(resultDict));

        var userStep = 0;
        var totalDistanceInMinutes = 0;
        var i = 0;

        Console.WriteLine();
        Console.WriteLine($"Choice # {count}");

        foreach (var item in result.objects)
        {
            i++;
            if (item.type.ToString() == "edge")
            {
                var distanceInMinutes = item.properties.distanceInMinutes.Value;
                totalDistanceInMinutes += distanceInMinutes;
                var edgeInfo = $"    ({item.label} = {distanceInMinutes} min)";
                Console.WriteLine(edgeInfo);
            }
            else
            {
                userStep++;
                var userStepCaption = (userStep == 1 ? "Start at" : (i == result.objects.Count ? "Arrive at" : "Go to")
                var vertexInfo = $"{userStep}. {userStepCaption} {item.label} = {item.id}";
```

```
118             foreach (var item in result.objects)
119             {
120                 i++;
121                 if (item.type.ToString() == "edge")
122                 {
123                     var distanceInMinutes = item.properties.distanceInMinutes.Value;
124                     totalDistanceInMinutes += distanceInMinutes;
125                     var edgeInfo = $"    ({item.label} = {distanceInMinutes} min)";
126                     Console.WriteLine(edgeInfo);
127                 }
128                 else
129                 {
130                     userStep++;
131                     var userStepCaption = (userStep == 1 ? "Start at" : (i == result.objects.Count ? "Arrive at" : "Go to")
132                     var vertexInfo = $"{userStep}. {userStepCaption} {item.label} = {item.id}";
133
134                     switch (item.label.ToString())
135                     {
136                         case "gate":
137                             vertexInfo += $", airline = {item.properties.airline[0].value}";
138                             break;
139
140                         case "restaurant":
141                             vertexInfo += $", rating = {item.properties.rating[0].value}";
142                             vertexInfo += $", avg price = {item.properties.averagePrice[0].value}";
143                             break;
144                     }
145
146                     vertexInfo += $" ({totalDistanceInMinutes} min)";
147                     Console.WriteLine(vertexInfo);
148                 }
149             }
150
```

```
118             foreach (var item in result.objects)
C:\Users\lenni\OneDrive\Projects\Demo Code\Pluralsight\CosmosDBDemos\Code\Module10-Gre...

*** Scenario 1 - First eat (> .3 rating), then switch terminals, then go to gate ***

Choice # 1
1. Start at gate = Gate T1-2, airline = Continental (0 min)
2. Go to restaurant = Wendys, rating = 0.4, avg price = 9.5 (2 min)
3. Go to terminal = Terminal 1 (7 min)
    (terminalToNextTerminal = 10 min)
4. Go to terminal = Terminal 2 (17 min)
    (terminalToGate = 7 min)
5. Arrive at gate = Gate T2-3, airline = Delta (24 min)

Choice # 2
1. Start at gate = Gate T1-2, airline = Continental (0 min)
2. Go to restaurant = Chipotle, rating = 0.6, avg price = 12.5 (6 min)
3. Go to terminal = Terminal 1 (16 min)
    (terminalToNextTerminal = 10 min)
4. Go to terminal = Terminal 2 (26 min)
    (terminalToGate = 7 min)
5. Arrive at gate = Gate T2-3, airline = Delta (33 min)

                    }

                    vertexInfo += $" ({totalDistanceInMinutes} min)";
                    Console.WriteLine(vertexInfo);
                }
            }
150
```



Thank You!

Contact me

- lenni.lobel@sleektech.com

Visit my blog

- lennilobel.wordpress.com

Follow me on Twitter

- @lennilobel

Thanks for coming! ☺