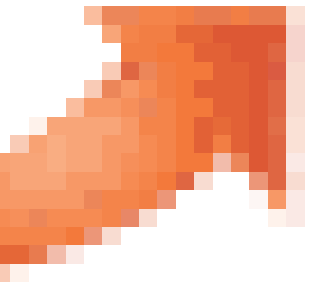
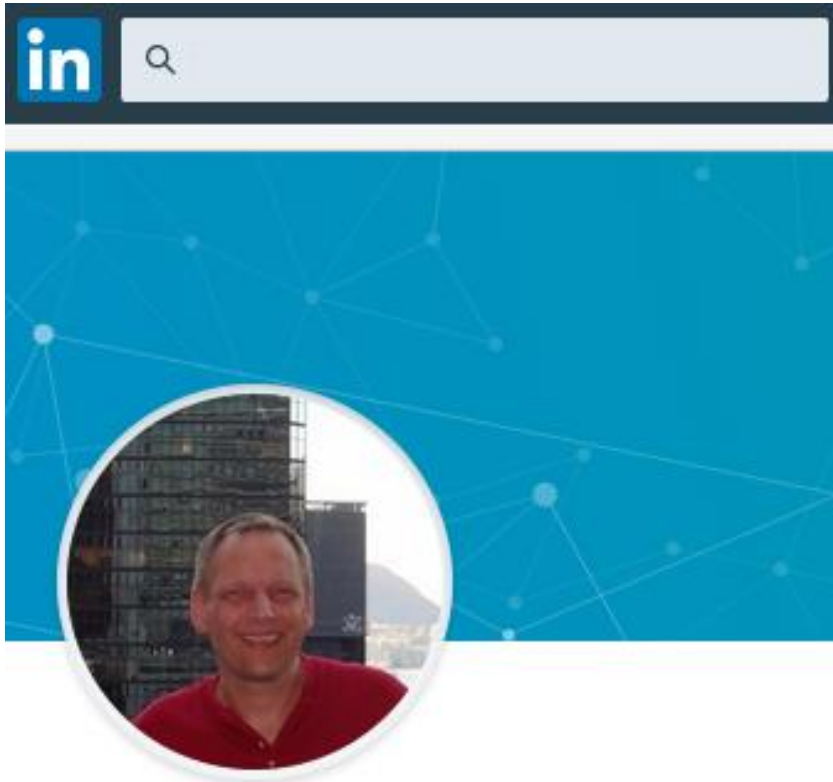


Cosmos DB

get started with NoSQL on Azure

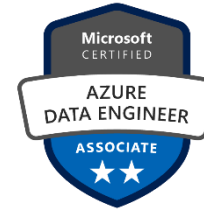


about me

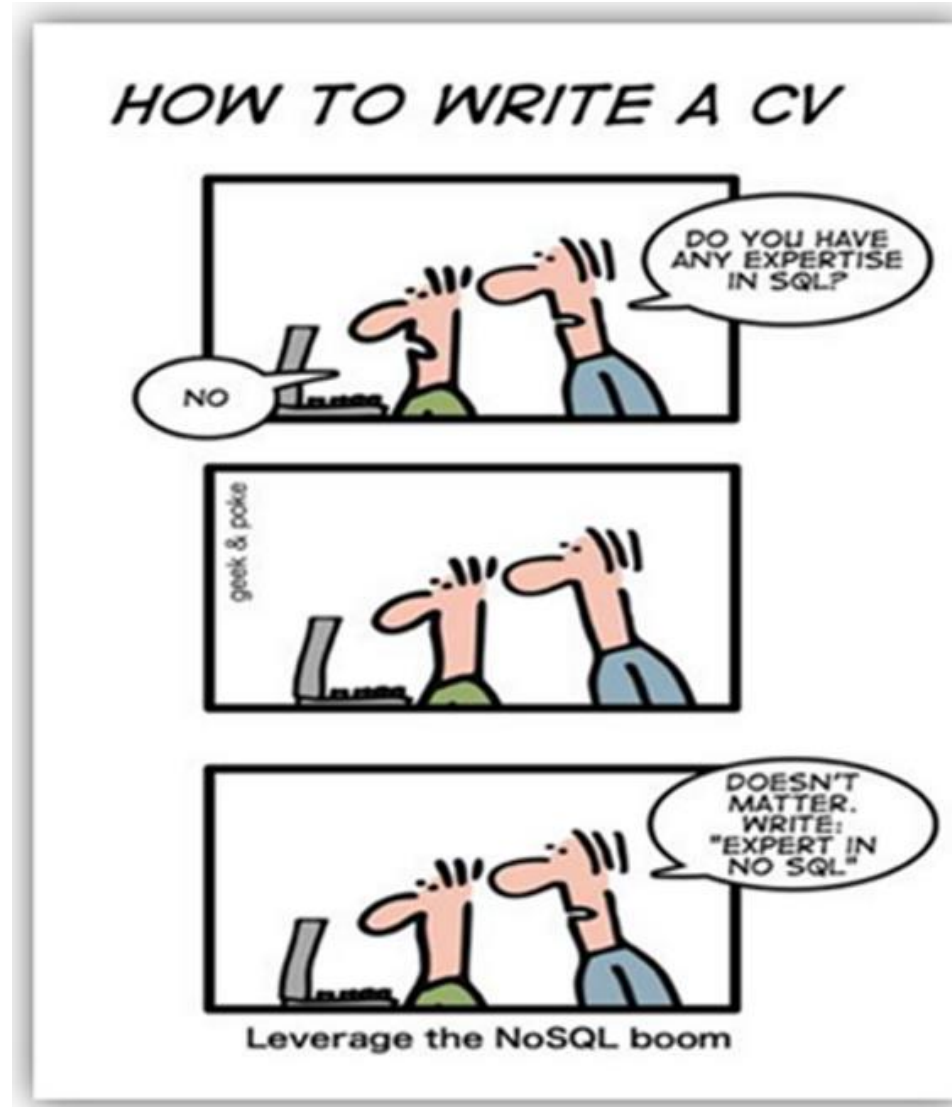


Alexander Karl

<https://www.linkedin.com/in/alexander-karl-44561012a/>



erste Berührungen ...



... und

RECENTLY DURING THE JOB INTERVIEW



ernsthafte Berührungen ...

Data Pipeline: The simplified data pipeline for this initiative is as shown in Figure 5.10:

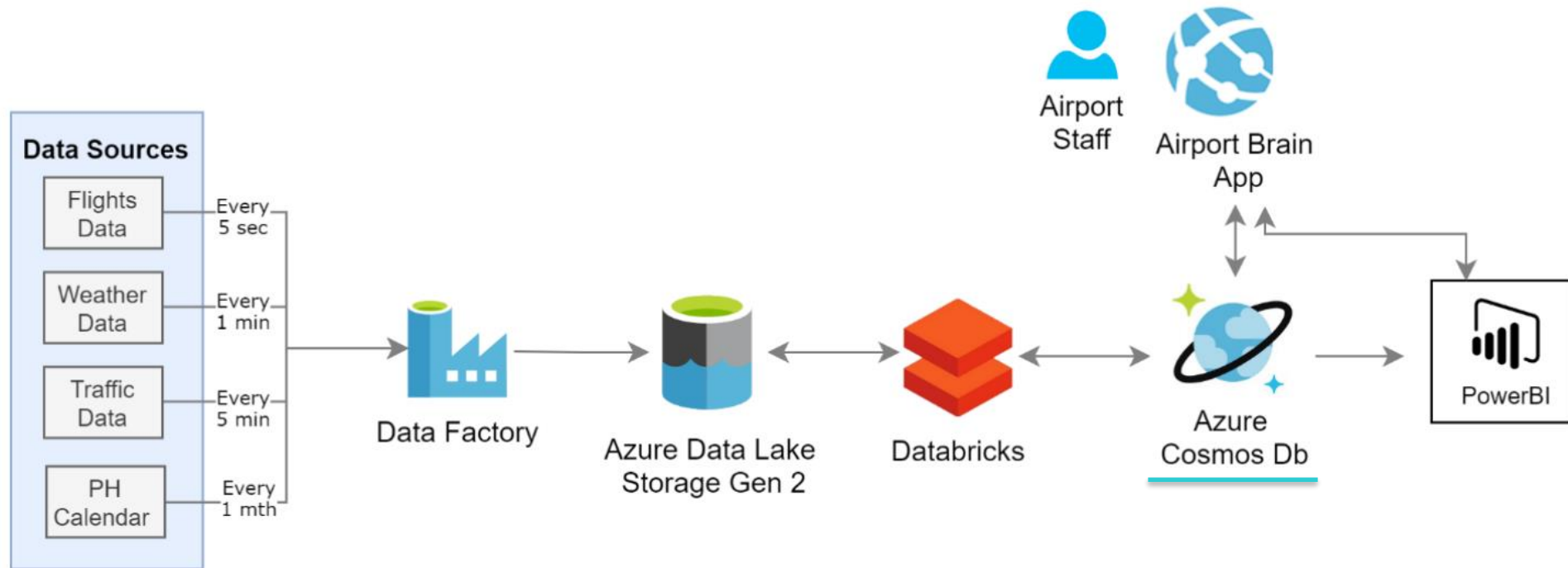
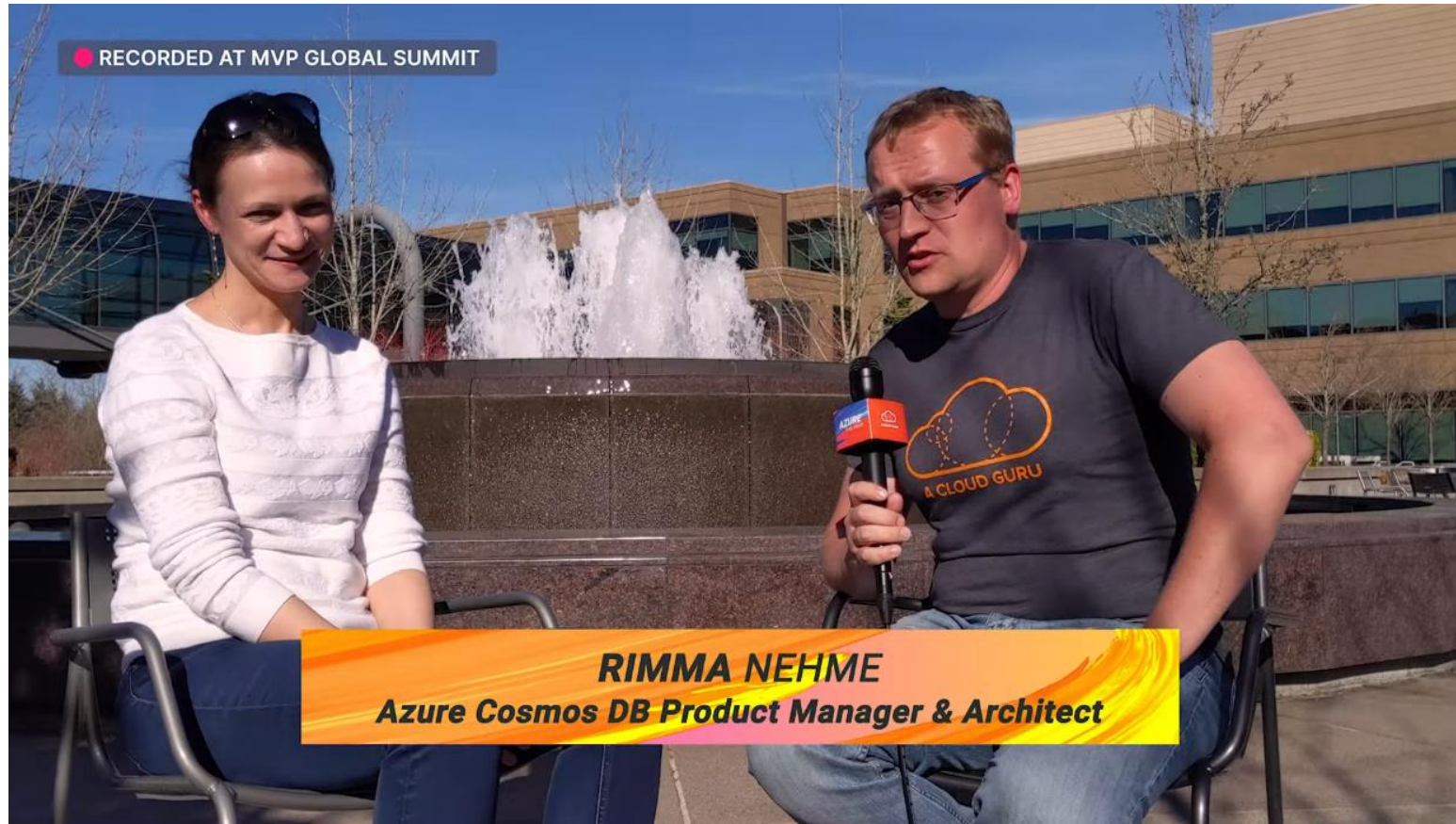


Figure 5.10: Data pipeline for initiative 2

... und



<https://www.youtube.com/watch?v=WCY3KU8XrmY>

Agenda:

NoSQL vs. RDBMS

Provision Cosmos DB

Demo Import & Query

NoSQL vs. Relational DBMS

Vereinfachte Anwendungsentwicklung

- schemalose Daten
- Open-Source-APIs
- mehrere SDKs

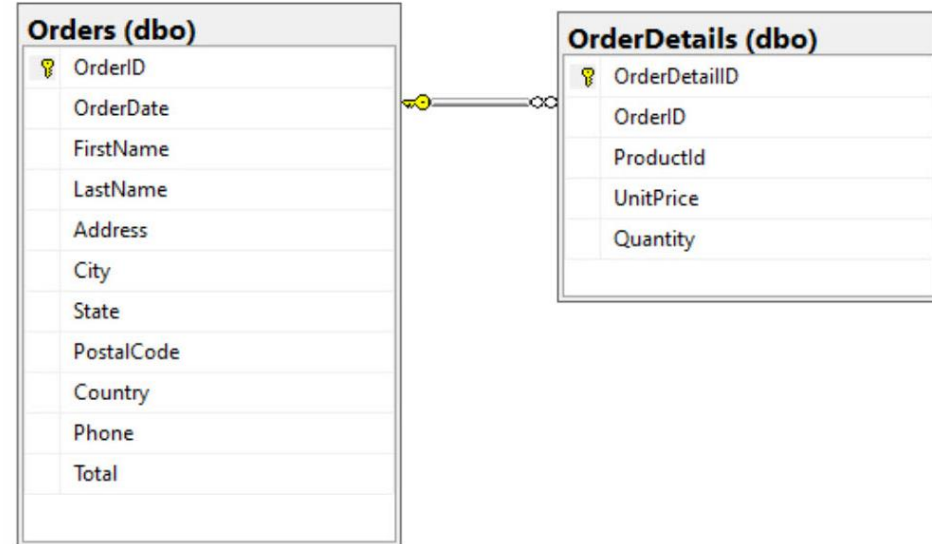
<https://docs.microsoft.com/de-de/azure/cosmos-db/relational-nosql>

<https://docs.microsoft.com/de-de/azure/cosmos-db/introduction>

NoSQL

vs. Relational DBMS

```
{
  "OrderId": 1,
  "OrderDate": 1574161910220,
  "FirstName": "John",
  "LastName": "Smith",
  "Address": "10 Street",
  "City": "City",
  "State": "VA",
  "OrderDetails": [
    {
      "UnitPrice": 7.99,
      "OrderDetailId": 2,
      "Quantity": 1,
      "ProductId": 259694,
      "OrderId": 1
    },
    {
      "UnitPrice": 7.99,
      "OrderDetailId": 3,
      "Quantity": 1,
      "ProductId": 295693,
      "OrderId": 1
    }
  ],
  "id": "795c50dc-1a83-11ea-bf07-00163ee85f66",
  "_rid": "VdgtAK230MANAAAAAAAAA==",
  "_self": "dbs/VdgtAA=/colls/VdgtAK230MA=/docs/VdgtAK230MANAAAAAAAAA==/",
  "_etag": "\"370017e1-0000-1100-0000-5df770f20000\"",
  "_attachments": "attachments/",
  "_ts": 1576497394
}
```



<https://docs.microsoft.com/de-de/azure/cosmos-db/relational-nosql>

5 different APIs

Azure Cosmos DB

Microsoft's globally distributed, massively scalable, multi-model database service



<https://azure.microsoft.com/de-de/blog/azure-cosmos-db-database-for-intelligent-cloud-intelligent-edge-era/>

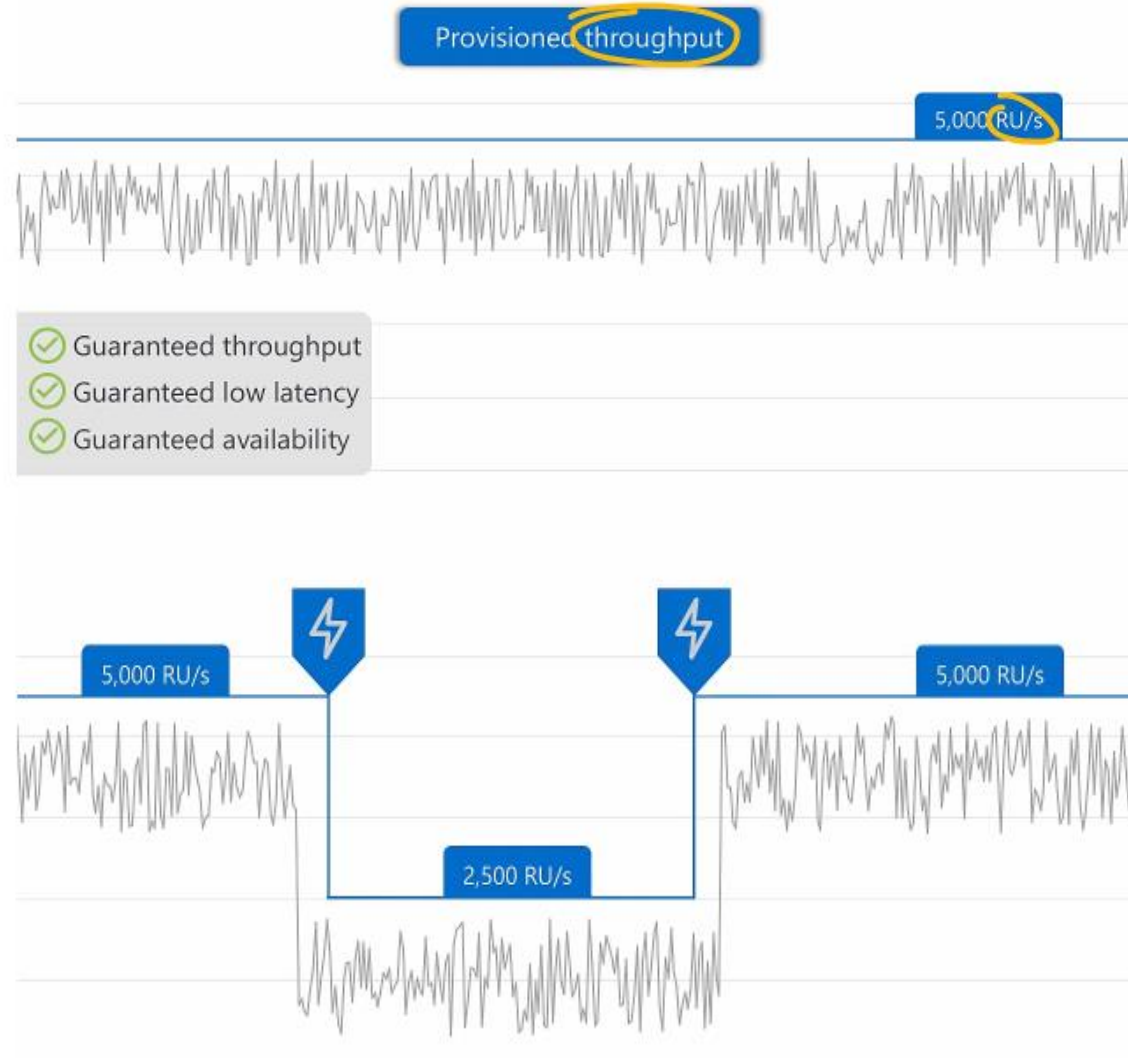
Capacity mode

Provisioned

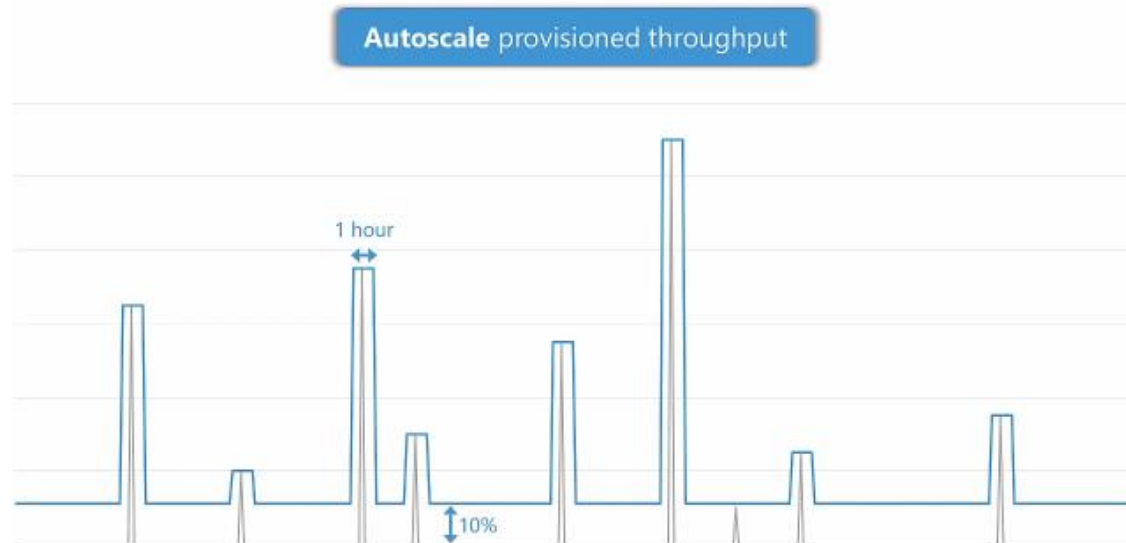
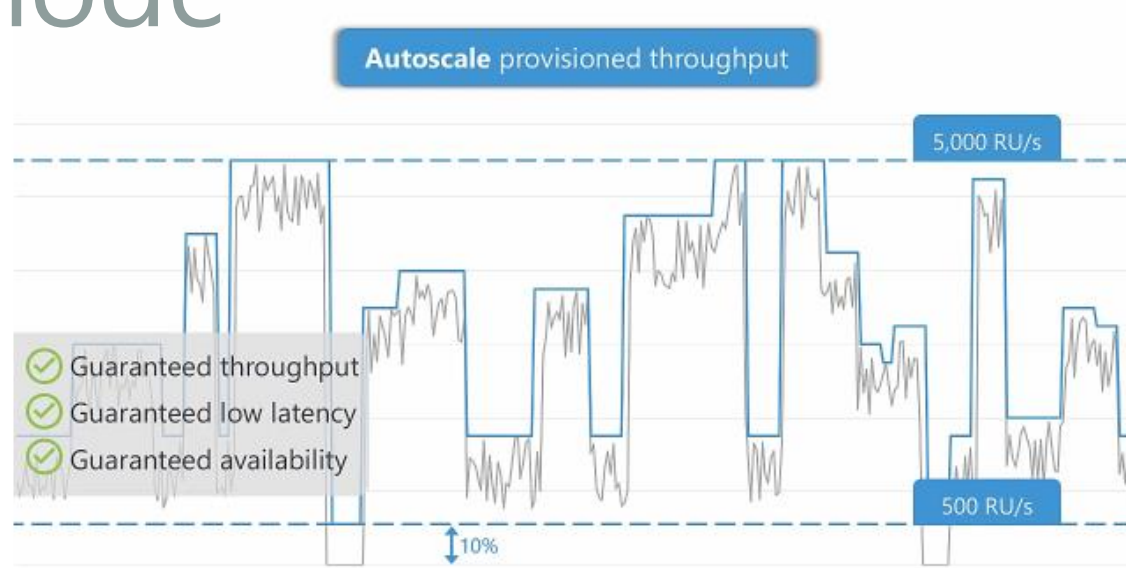
- Standard
- Autoscale

Serverless

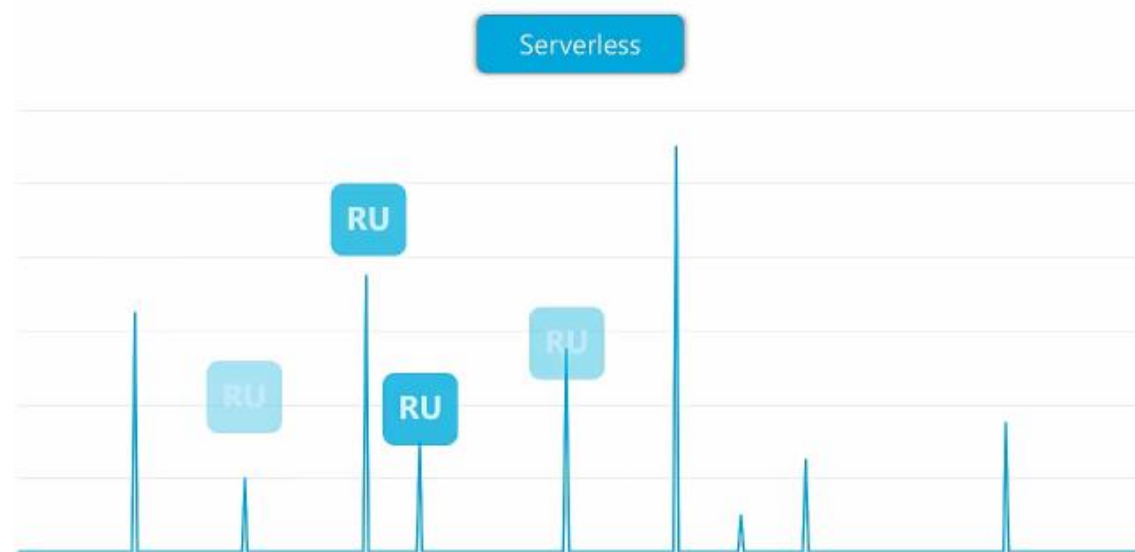
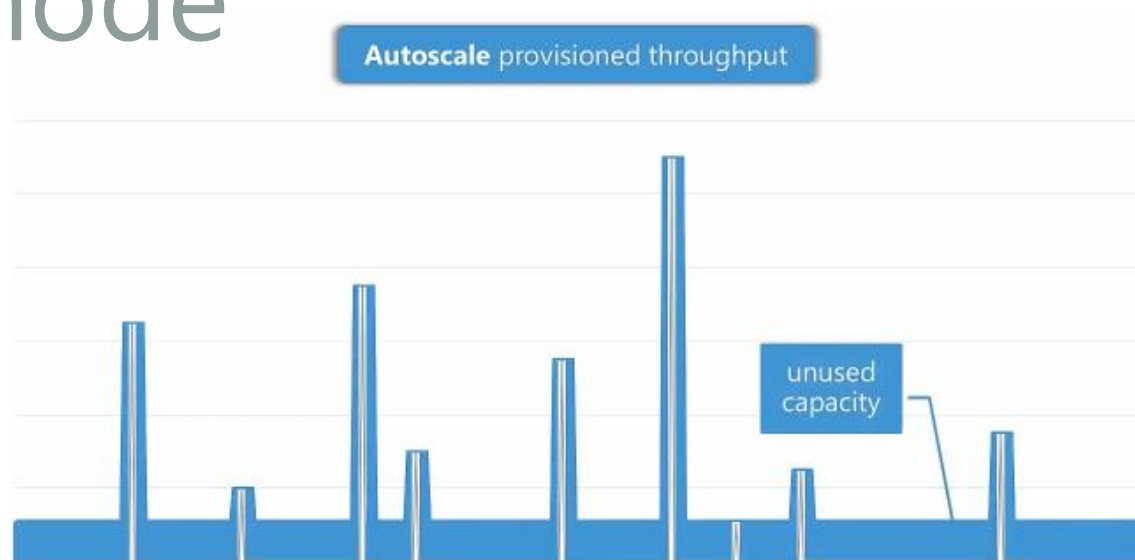
Capacity mode



Capacity mode



Capacity mode

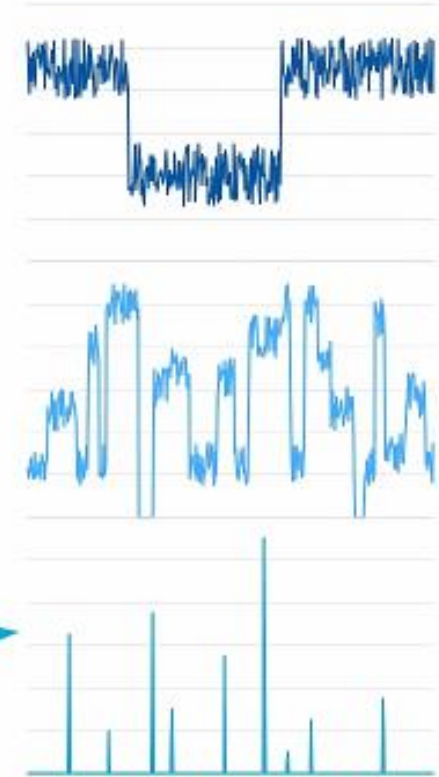
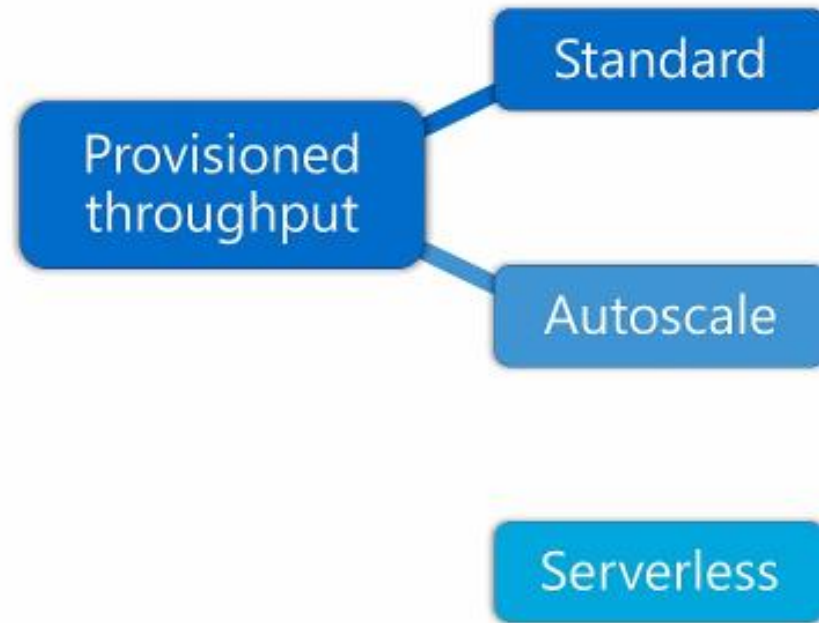


Capacity mode

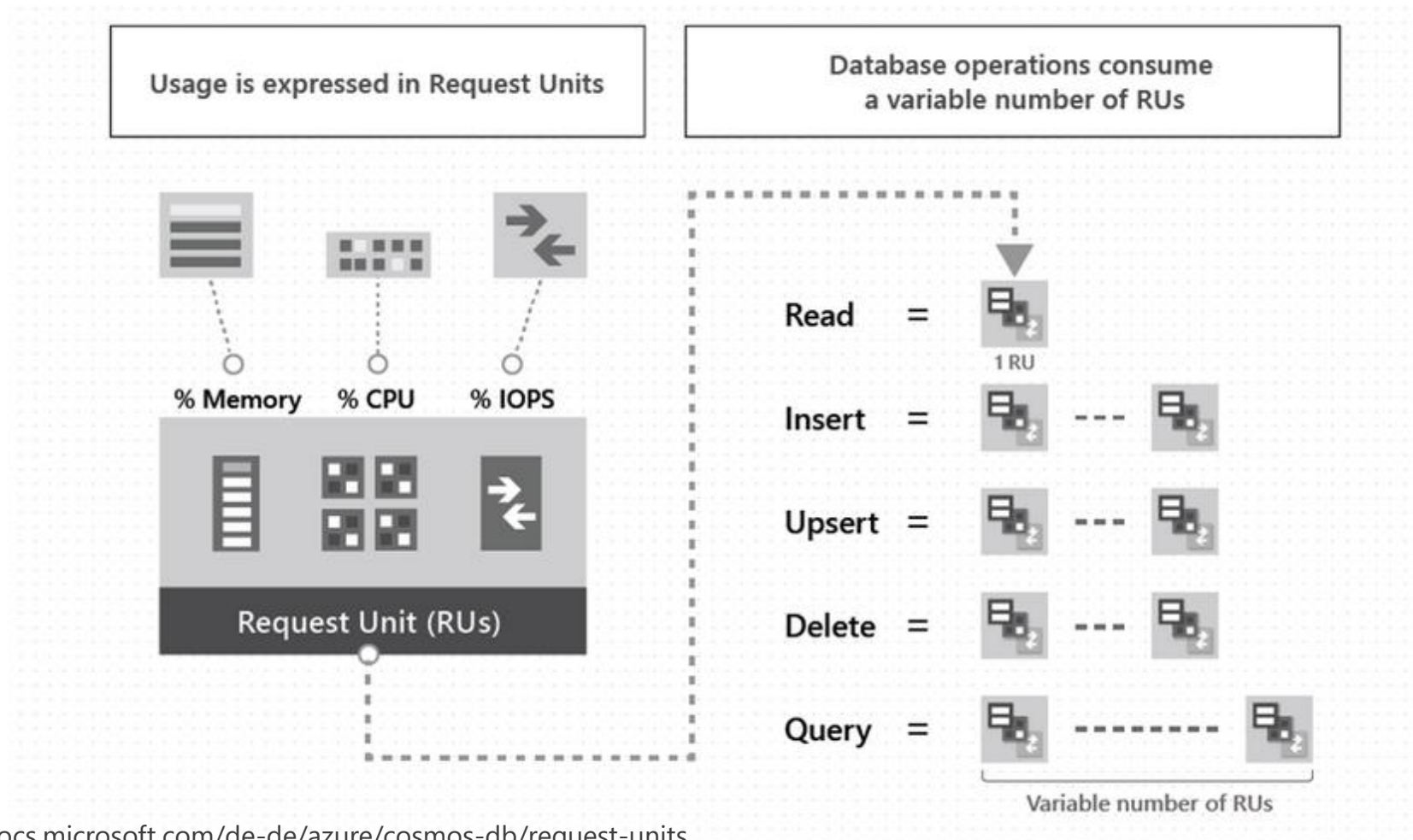
Provisioned

- Standard
- Autoscale

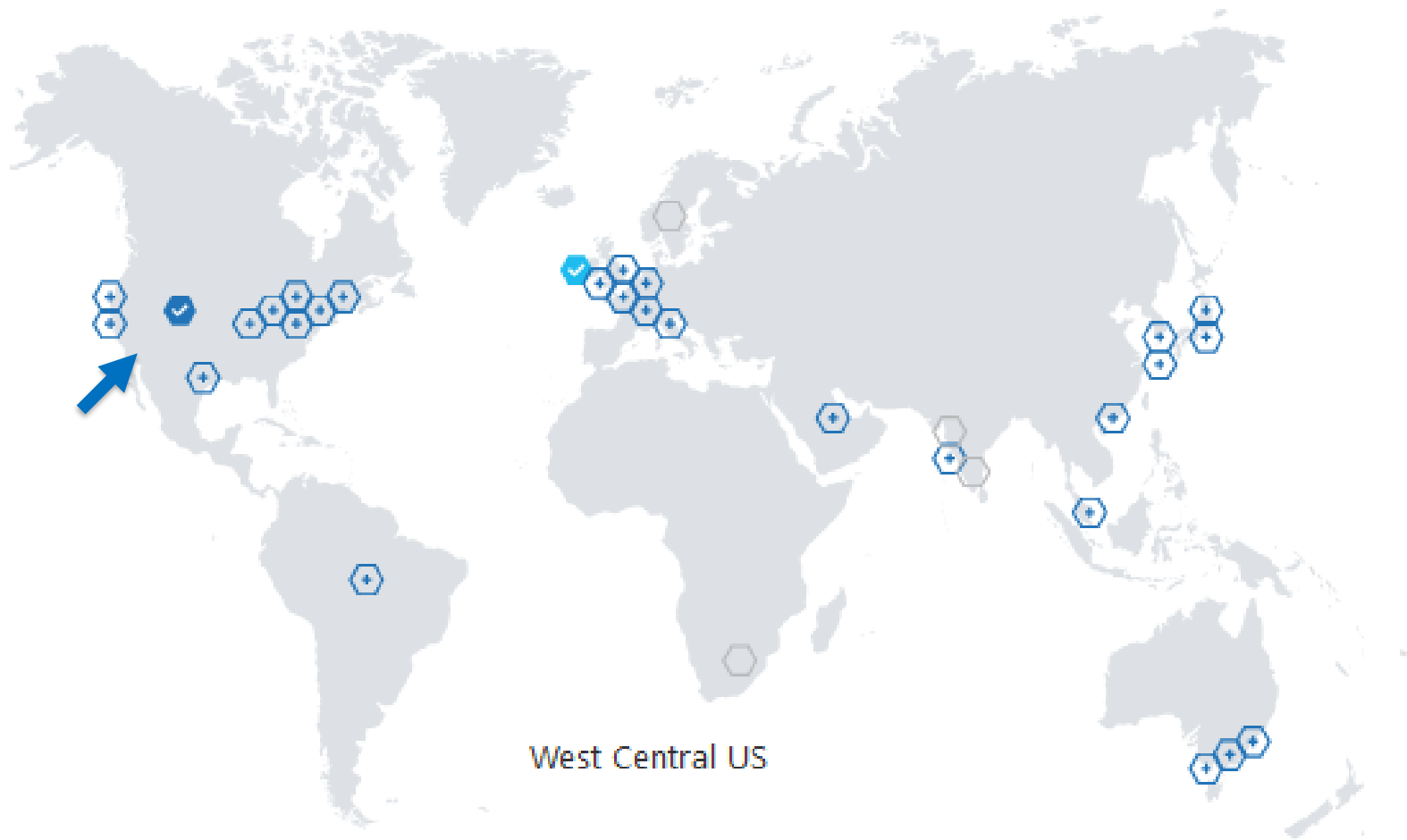
Serverless



Request-Units RU

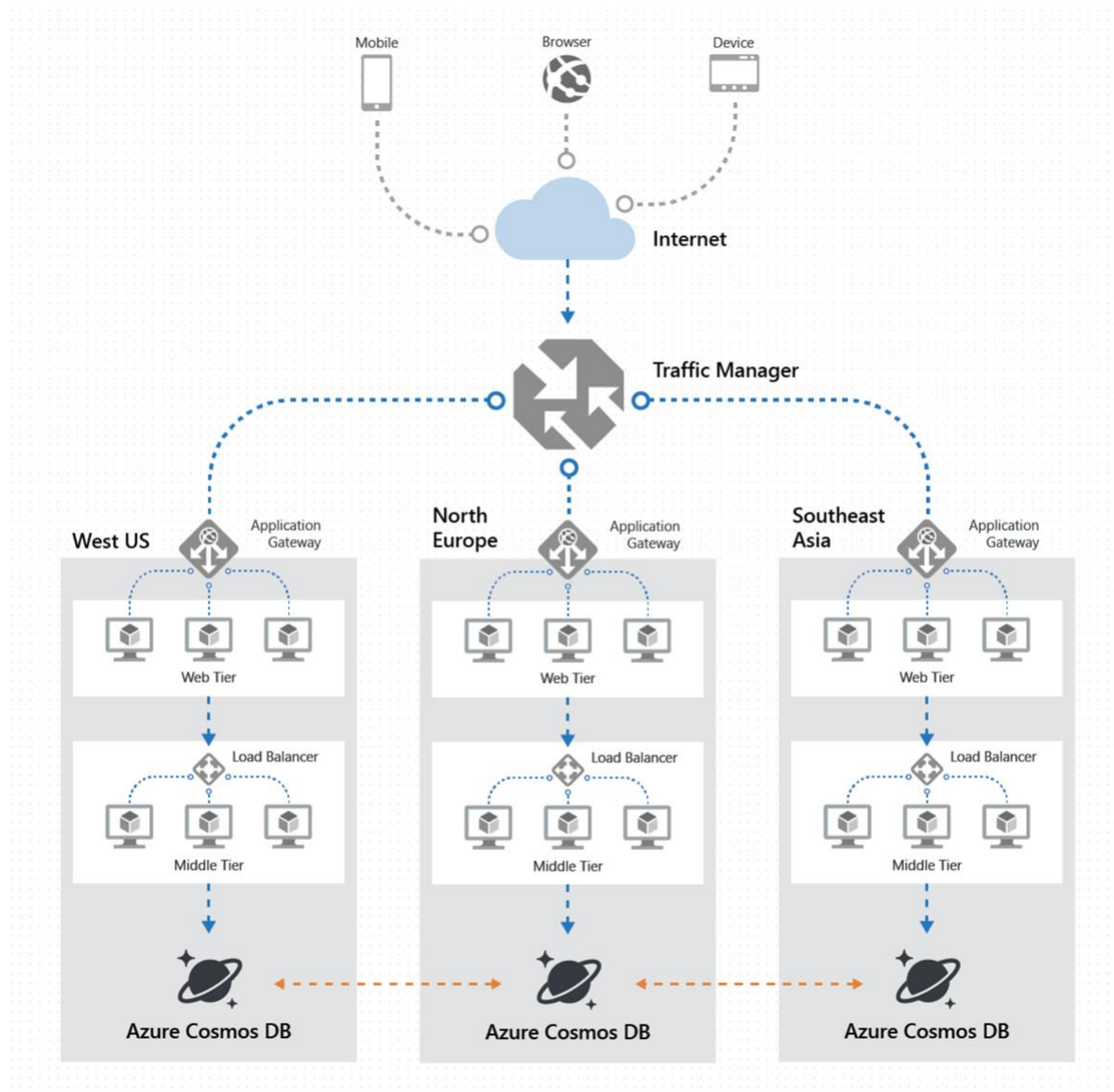


Replicate data globally



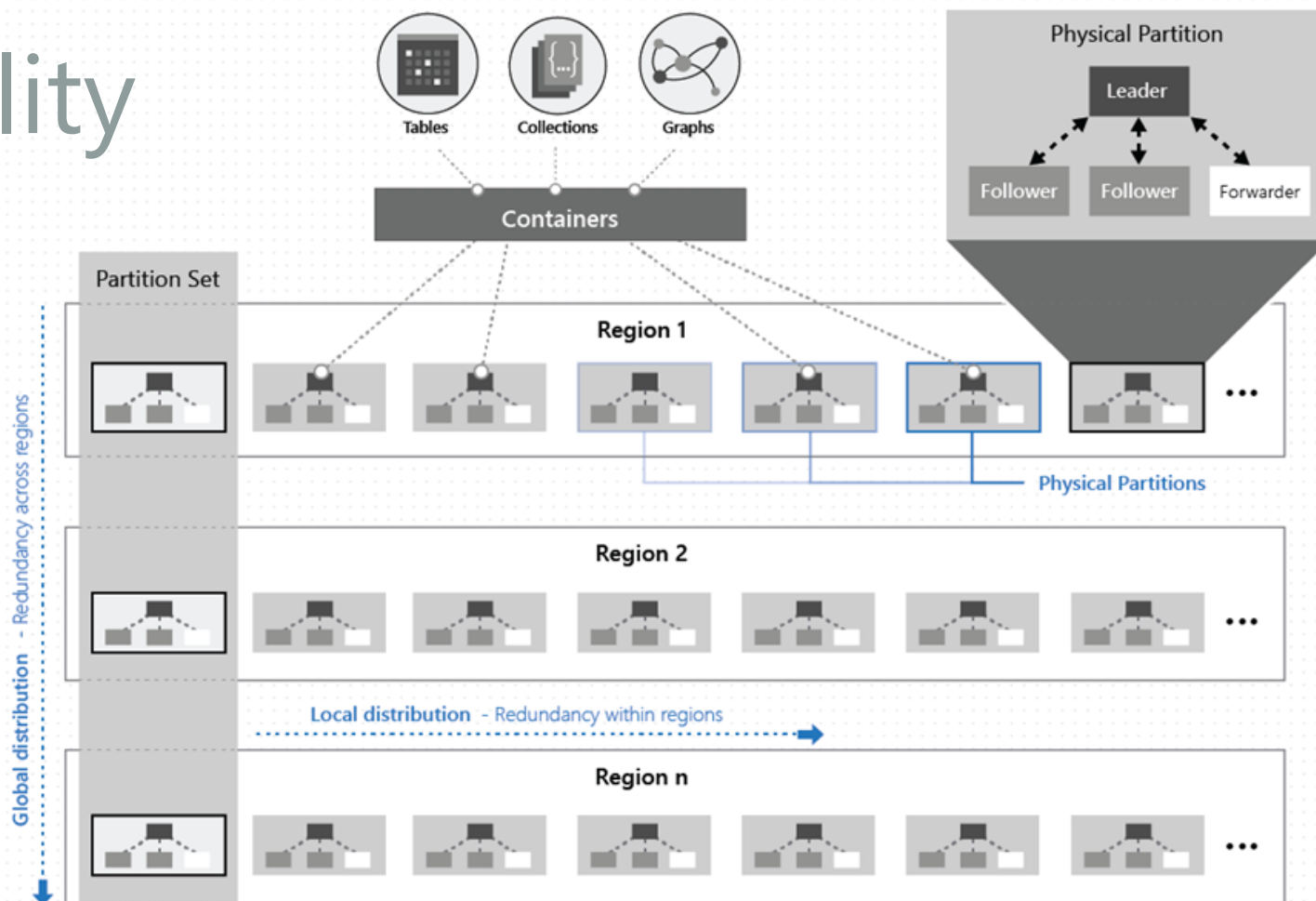
West Central US

Distribute data globally

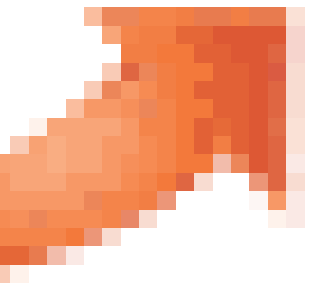


<https://docs.microsoft.com/de-de/azure/cosmos-db/distribute-data-globally>

High availability



Demo



Demo



Azure Cosmos DB Data Migration Tool

Target Information

Welcome	Specify target information
Source Information	
Target Information	Export to: Azure Cosmos DB - Sequential record import
Advanced	
Summary	Connection String
Results	E9zlybSKeGekOr6rfdPING4zVjflUfKGpfG9sp1c
	Collection



Power BI Desktop

Comparing NoSQL vs. RDBMS

Document Database

Collections

Properties

Documents

Denormalized

Schema-less

No referential integrity

Relational Database

Tables

Columns

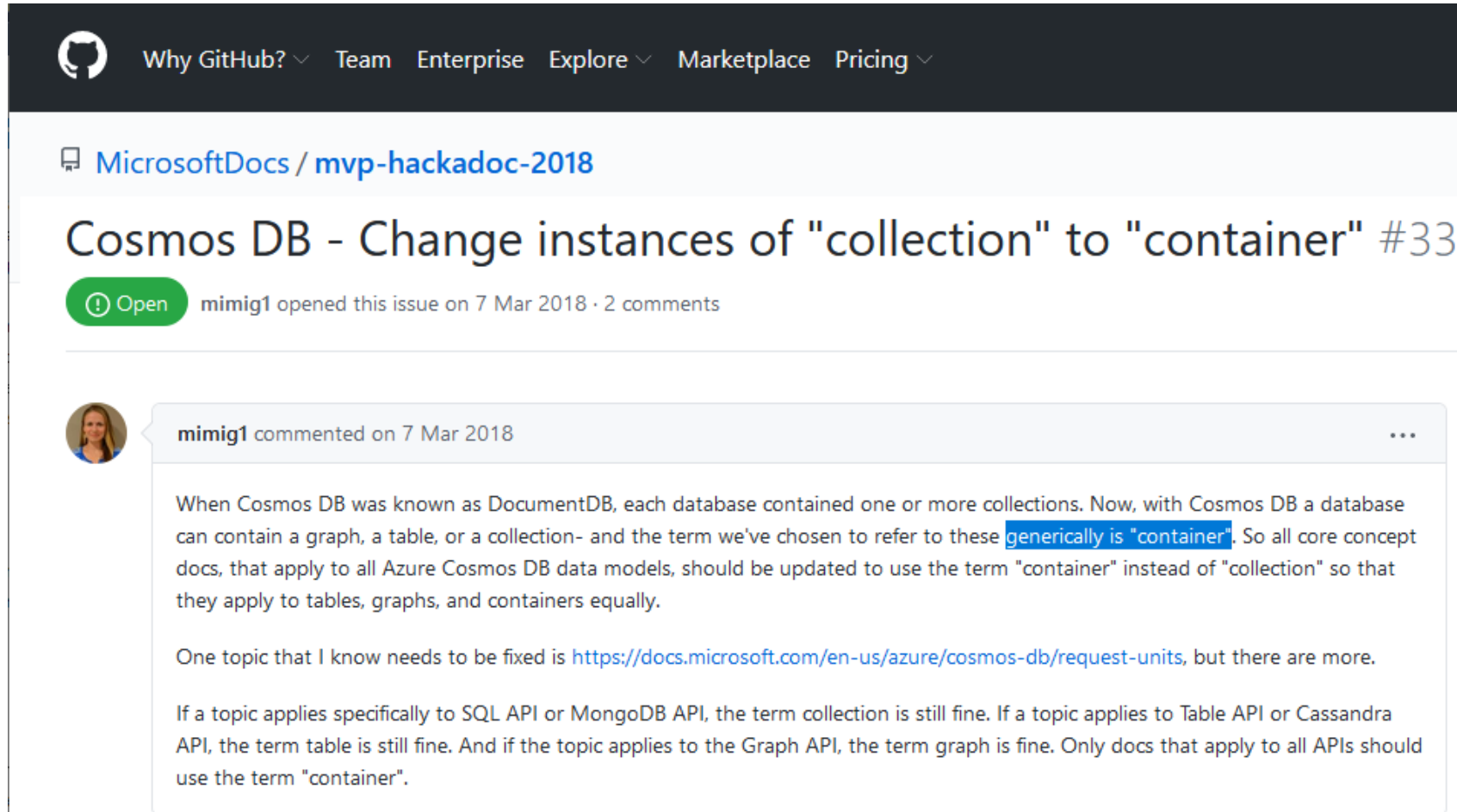
Rows

Normalized

Declarative Schema

Referential integrity

„Collections“ and „Containers“



Why GitHub? Team Enterprise Explore Marketplace Pricing

MicrosoftDocs / mvp-hackadoc-2018

Cosmos DB - Change instances of "collection" to "container" #33

Open mimig1 opened this issue on 7 Mar 2018 · 2 comments

mimig1 commented on 7 Mar 2018

When Cosmos DB was known as DocumentDB, each database contained one or more collections. Now, with Cosmos DB a database can contain a graph, a table, or a collection- and the term we've chosen to refer to these **generally is "container"**. So all core concept docs, that apply to all Azure Cosmos DB data models, should be updated to use the term "container" instead of "collection" so that they apply to tables, graphs, and containers equally.

One topic that I know needs to be fixed is <https://docs.microsoft.com/en-us/azure/cosmos-db/request-units>, but there are more.

If a topic applies specifically to SQL API or MongoDB API, the term collection is still fine. If a topic applies to Table API or Cassandra API, the term table is still fine. And if the topic applies to the Graph API, the term graph is fine. Only docs that apply to all APIs should use the term "container".

<https://github.com/MicrosoftDocs/mvp-hackadoc-2018/issues/33>

System properties

```
1  {
2      "id": "1",
3      "_rid": "9cN4ALEU-KcCAAAAAAAAAA==",
4      "_self": "dbs/9cN4AA==/colls/9cN4ALEU-Kc=/docs/9cN4ALEU-KcCAAAAAAAAAA==/",
5      "_etag": "\"0200b17e-0000-0c00-0000-600f46240000\"",
6      "_attachments": "attachments/",
7      "_ts": 1611613732
8  }
```

_rid = „Container ID“
_self = „individual address“
_etag = „unique identifier“
_ts = „timestamp“ since 1.1.1970

SQL Statements

The screenshot shows the SQL API interface for a database named 'VerlagDemo'. The left sidebar shows a tree view with 'Sales' selected. The main area displays a query window titled 'Query 1' with the following SQL statement:

```
SELECT s.VerkaufDatum
      , s.VerkaufStueck
      , s.Vertreter.ID
      , s.Vertreter.Name
FROM Sales s
join s.Vertreter v
```

Below the query, there are tabs for 'Results' and 'Query Stats'. The 'Results' tab is active, showing a list of 5 results. The first result is expanded, showing the following JSON data:

```
{
  "VerkaufDatum": "2009-05-07T00:00:00.0000000",
  "VerkaufStueck": 23,
  "ID": 1,
  "Name": "Wabnegg"
},
{
  "VerkaufDatum": "2009-05-14T00:00:00.0000000",
  "VerkaufStueck": 20,
  "ID": 1,
  "Name": "Wabnegg"
},
```

SQL Statements

SQL API

- VerlagDemo
 - Scale
 - Sales
 - Items
 - Settings
 - Stored Procedures
 - User Defined Functions
 - Triggers

Scale | Items | Query 1 | Query 2 ×

```
1 SELECT s.VerkaufDatum
2     , s.VerkaufStueck
3     , s.Buch.Buchtitel
4     , a.Name      as Autorennamen
5     , s.Buch.Preis
6     , b.Preis * s.VerkaufStueck as Umsatz
7 FROM Sales s
8 join s.Buch b
9 join s.Buch.Autor a
10 WHERE Contains (s.Buch.Buchtitel, "Windows")
```

Results Query Stats

1 - 5

```
{
  "VerkaufDatum": "2009-05-07T00:00:00.0000000",
  "VerkaufStueck": 23,
  "Buchtitel": "Konfigurieren von Windows Server 2008 Active Directory",
  "Autorennamen": "Holme",
  "Preis": 79,
  "Umsatz": 1817
},
{
  "VerkaufDatum": "2009-05-14T00:00:00.0000000",
  "VerkaufStueck": 20,
  "Buchtitel": "Konfigurieren von Windows Server 2008 Active Directory",
  "Autorennamen": "Holme",
  "Preis": 79,
  "Umsatz": 1580
},
```

Modeling data

Wann Sie einbetten sollten

Verwenden Sie in der Regel eingebettete Datenmodelle in den folgenden Fällen:

- Zwischen Entitäten gibt es **contained** -Beziehungen.
- Zwischen Entitäten gibt es **eins-zu-viele** -Beziehungen.
- Es gibt eingebettete Daten, die sich **selten ändern**.
- Es gibt eingebettete Daten, die nicht **grenzenlos** wachsen.
- Es gibt eingebettete Daten, die **häufig gemeinsam abgefragt** werden.

ⓘ Hinweis

In der Regel bieten denormalisierte Datenmodelle eine bessere **Leseleistung** .

stored-procedures

<https://github.com/Azure/azure-cosmosdb-js-server/blob/master/samples/stored-procedures/sum.js>



Why GitHub? Team Enterprise Explore Marketplace Pricing Search / Sign in Sign up

Azure / azure-cosmosdb-js-server

Watch

Code Issues 29 Pull requests 5 Actions Projects Security Insights

master azure-cosmosdb-js-server / samples / stored-procedures / sum.js / Jump to

Go to file



aliyu Add samples for unique constraint (as a pre-trigger) and sum (as a st... ..

Latest commit a7a2db8 on 14 Jun 2016 History

1 contributor

84 lines (71 sloc) 3.88 KB

Raw Blame



```
1 /**
2  * This is executed as a stored procedure to compute the sum of a specified feature in a collection.
3  * To avoid script timeout on the server when there are lots of documents (100K+), the script is executed in batches,
4  * each batch sums the value of the specified feature in the batch docs and returns continuation token.
5  * The script is run multiple times, starting from empty continuation,
6  * then using continuation returned by last invocation script until continuation returned by the script is null/empty string.
7  *
8  * @param {String} feature - Feature to be aggregated (required).
9  * @param {String} filterQuery - Optional filter for query (e.g. "SELECT * FROM docs WHERE docs.category = 'food'").
10 * @param {String} continuationToken - The continuation token passed by request, continue counting from this token.
11 */
12 function sum(feature, filterQuery, continuationToken) {
13
14     const ERROR_CODES = {
15         BAD_REQUEST: 400,
16         NOT_FOUND: 404,
17         CONFLICT: 409,
18         RETRY_WITH: 449,
19         NOT_ACCEPTED: 499
20     };
```

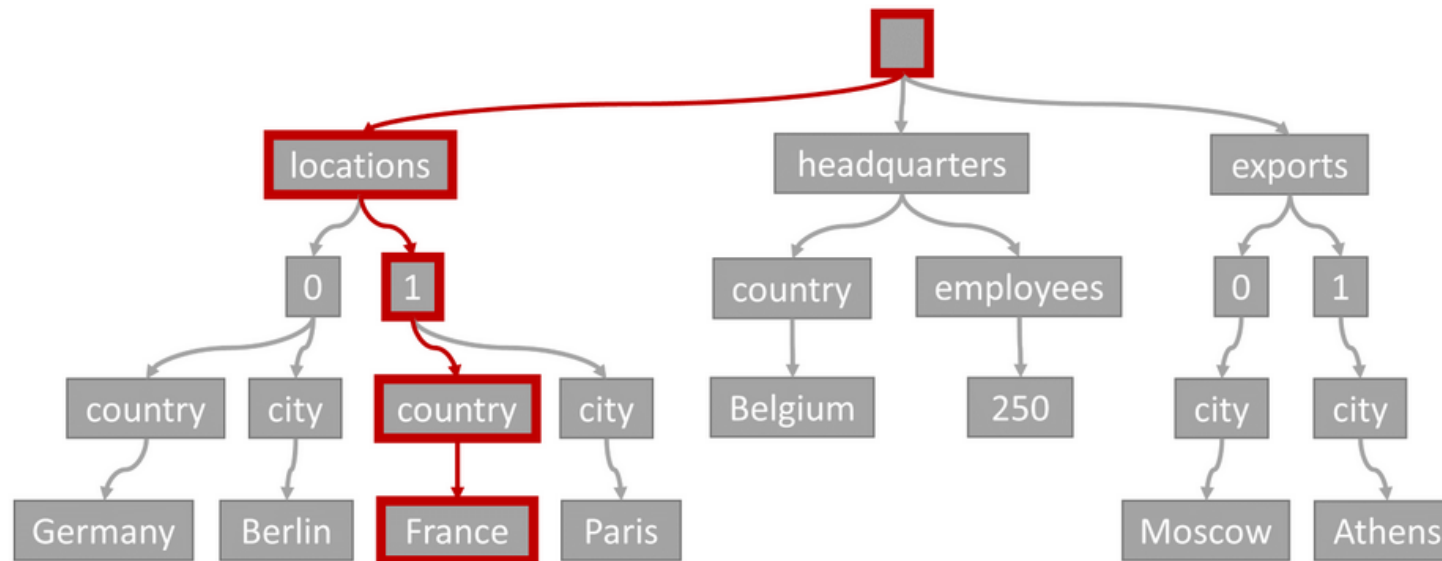
<https://github.com/Azure/azure-cosmosdb-js-server/blob/master/samples/stored-procedures/sum.js>

Index-overview

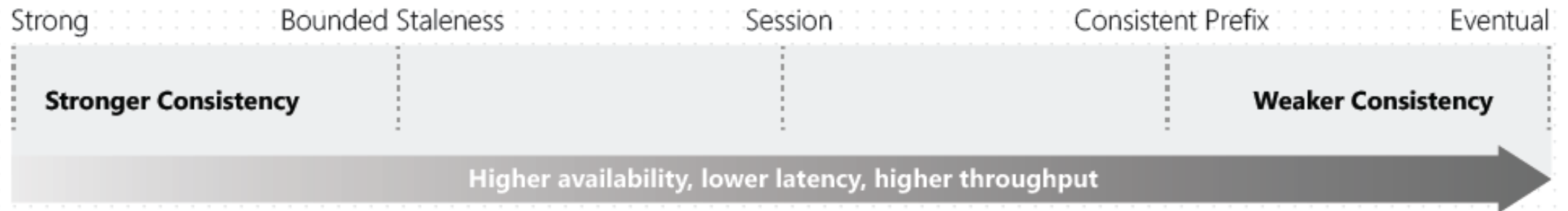
Abfragen mit Indizes

Die während der Indizierung der Daten extrahierten Pfade vereinfachen das Suchen im Index bei der Verarbeitung einer Abfrage. Durch einen Abgleich der `WHERE`-Klausel einer Abfrage mit der Liste der indizierten Pfade ist es möglich, sehr schnell die Elemente zu ermitteln, die dem Abfrageprädikat entsprechen.

Betrachten Sie beispielsweise die folgende Abfrage: `SELECT location FROM location IN company.locations WHERE location.country = 'France'`. Das Abfrageprädikat (nach Elementen filtern, die an beliebiger Stelle „France“ als Land oder Region enthalten) würde dem Pfad entsprechen, der rot hervorgehoben ist:



Consistency levels



Comparing NoSQL vs. RDBMS

Document Database

BASE

Basic Availability

- The database appears to work most of the time.

Soft-state

- Stores don't have to be write-consistent, nor do different replicas have to be mutually consistent all the time.

Eventual consistency

- Stores exhibit consistency at some later point (e.g., lazily at read time).

Relational Database

ACID

Atomic

- All operations in a transaction succeed or every operation is rolled back.

Consistent

- On the completion of a transaction, the database is structurally sound.

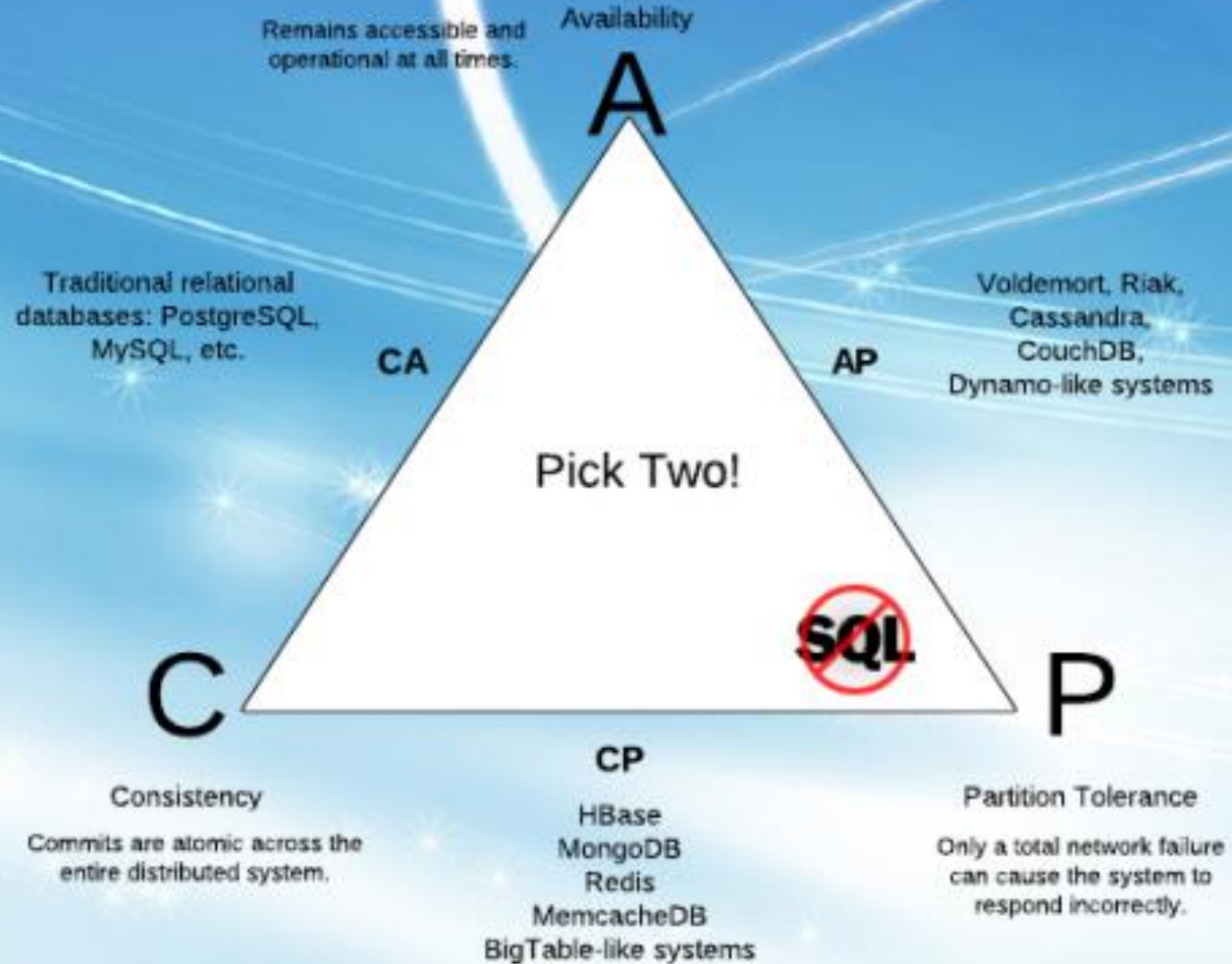
Isolated

- Transactions do not contend with one another. Contentious access to data is moderated by the database so that transactions appear to run sequentially.

Durable

- The results of applying a transaction are permanent, even in the presence of failures.

Scalability: CAP Theorem



Tools

Data migration tool

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

Azure Storage Explorer (SQL + Table)

Robo 3T (mongo)

MongoDB Compass

Quick start

Quick start

Congratulations! Your Azure Cosmos DB account was created.

Now, let's connect to it using a sample app:

Choose a platform

[.NET](#) [Xamarin](#) [Java](#) [Node.js](#) [Python](#)

Choose to setup with either notebook or download .NET app

A Add container & work with data using notebook

You can get started with Cosmos DB for a simple one click setup using notebook. A sample container, .NET app will be setup for you to query and edit your data.

[Setup with notebook](#)

B Step 1: Add a container

"Items" container has been created with 10GB storage capacity and 400 [Request Units per second \(RU/s\)](#) throughput capacity, for up to 400 reads/sec. Estimated hourly bill: \$0.033 USD

Step 2: Download and run your .NET app

We created a sample .NET app connected to your "Items" container. Download, extract, build and run the app.

[Download](#)

Step 3: Work with data

Query and edit your data, add stored procedures, and more using Data Explorer.

[Open Data Explorer](#)

<https://docs.microsoft.com/de-de/azure/cosmos-db/sql-api-sdk-dotnet>

Quick start

The screenshot displays the Visual Studio IDE with a C# application running in the debugger. The code in the background is for a Cosmos DB tutorial, showing methods like `ReplaceFamilyItemAsync` and `DeleteFamilyItemAsync`. A console window in the foreground shows the application's output, including database creation, item updates, and deletion. The Diagnostic Tools window on the right shows performance metrics like Process Memory (34 MB) and CPU Usage.

```
Process: [5044] dotnet.exe
Program.cs App.config
CosmosGettingStarted (netcoreapp2.1)
CosmosGettingStartedTutorial.Program
DeleteFamilyItemAsync()

248
249
250
251
252 // </QueryItemsAsync>
253
254 // <ReplaceFamilyItemAsync>
255 /// <summary>
256 /// Replace an item in the container
257 /// </summary>
258 private async Task ReplaceFamilyItemAsync()
259 {
260     ItemResponse<Family> wakefieldFamilyResponse = await this.container.GetItemAsync<Family>(familyId, new PartitionKey(partitionKeyValue));
261     var itemBody = wakefieldFamilyResponse.Resource;
262
263     // update registration status from false to true
264     itemBody.IsRegistered = true;
265     // update grade of child
266     itemBody.Children[0].Grade = 6;
267
268     // replace the item with the updated content
269     wakefieldFamilyResponse = await this.container.ReplaceItemAsync<Family>(familyId, new PartitionKey(partitionKeyValue), itemBody);
270     Console.WriteLine("Updated Family [{0},{1}]\n", familyId, itemBody.Children[0].Grade);
271 }
272 // </ReplaceFamilyItemAsync>
273
274 // <DeleteFamilyItemAsync>
275 /// <summary>
276 /// Delete an item in the container
277 /// </summary>
278 private async Task DeleteFamilyItemAsync()
279 {
280     var partitionKeyValue = "Wakefield";
281     var familyId = "Wakefield.7";
282
283     // Delete an item. Note we must provide the partition key
284     ItemResponse<Family> wakefieldFamilyResponse = await this.container.DeleteItemAsync<Family>(familyId, new PartitionKey(partitionKeyValue));
285     Console.WriteLine("Deleted Family [{0},{1}]\n", partitionKeyValue, familyId);
286 }
287 // </DeleteFamilyItemAsync>
```

```
C:\Program Files\dotnet\dotnet.exe
Beginning operations...
Created Database: ToDoList
Created Container: Items
Current provisioned throughput : 800
New provisioned throughput : 900
Item in database with id: Andersen.1 already exists
Item in database with id: Wakefield.7 already exists
Running query: SELECT * FROM c WHERE c.LastName = 'Andersen'
Read {"id":"Andersen.1","LastName":"Andersen","Parents":[{"FamilyName":null,"FirstName":"Thomas"}, {"FamilyName":null,"FirstName":"Mary Kay"}],"Children":[{"FamilyName":null,"FirstName":"Henriette Thaulow","Gender":"female","Grade":5,"Pets":[{"GivenName":"Fluffy"}]}],"Address":{"State":"WA","County":"King","City":"Seattle"},"IsRegistered":false}
Updated Family [Wakefield,Wakefield.7].
Body is now: {"id":"Wakefield.7","LastName":"Wakefield","Parents":[{"FamilyName":"Wakefield","FirstName":"Robin"}, {"FamilyName":"Miller","FirstName":"Ben"}],"Children":[{"FamilyName":"Merriam","FirstName":"Jesse","Gender":"female","Grade":6,"Pets":[{"GivenName":"Goofy"}, {"GivenName":"Shadow"}]}],"FamilyName":"Miller","FirstName":"Lisa","Gender":"female","Grade":1,"Pets":null},"Address":{"State":"NY","County":"Manhattan","City":"NY"},"IsRegistered":true}
End of demo, press any key to exit.
```

Diagnostic Tools
Diagnostics session: 14 seconds
Events
Process Memory (MB)
CPU (% of all processors)

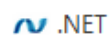
Quick start

Quick start

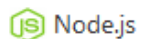
Congratulations! Your Azure Cosmos DB for MongoDB API account is ready.

Now, let's connect your existing MongoDB app to it:

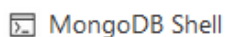
Choose a platform



.NET



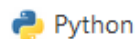
Node.js



MongoDB Shell



Java



Python

Others

Connect your existing MongoDB app

Use the host and password below to connect to your new account. Stay secure by using TLS 1.2.

HOST

cosmos01.mongo.cosmos.azure.com

PORT

10255

USERNAME

ibrik-cosmos01

PRIMARY PASSWORD

h8vf9BtruBlWyZmB2yUpSA0gj0bcC8nm2eUg

8Ohwkj7wKkvOyMWaoKg==

Check your driver specific SSL configuration in [this article](#).

Questions? [Contact us](#)

some more links

<https://github.com/movcom/mikepfeifferieapp-documentdb>

<https://docs.microsoft.com/de-de/azure/architecture/guide/technology-choices/data-store-overview>

<https://github.com/Azure/azure-cosmos-dotnet-v2/tree/master/samples>

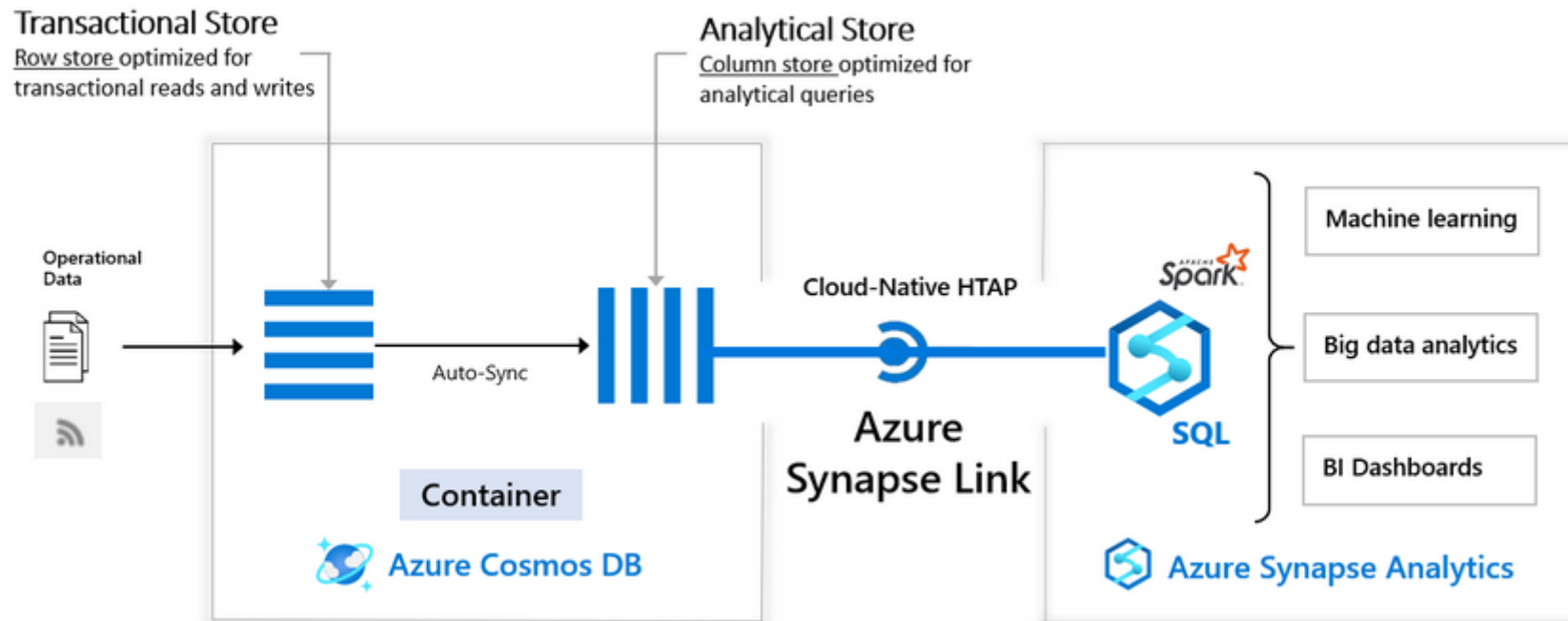
<https://azure.microsoft.com/de-de/blog/announcing-general-availability-of-azure-cosmos-db-reserved-capacity/>

Azure Synapse Link

<https://docs.microsoft.com/de-de/azure/cosmos-db/synapse-link>

<https://docs.microsoft.com/de-de/azure/cosmos-db/analytical-store-introduction>

<https://docs.microsoft.com/de-de/azure/cosmos-db/synapse-link-frequently-asked-questions>



Sponsors

You Rock! Sponsor



Gold Sponsor



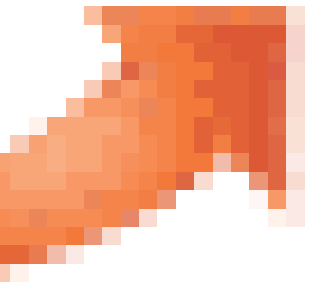
Silver Sponsor



Bronze Sponsor



.. bis zum nächsten event





.. jetzt einen schönen
Feierabend