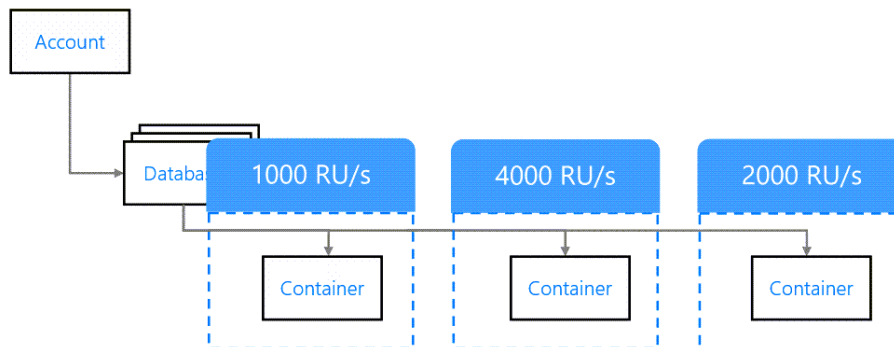


Module 02: Plan and implement Azure Cosmos DB SQL API

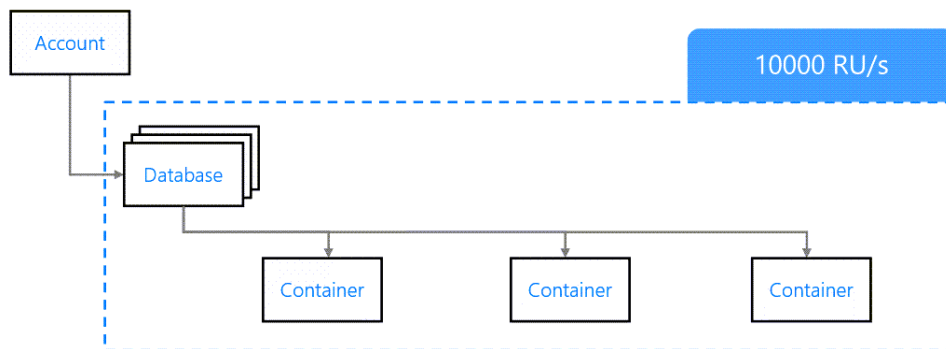
Plan Resource Requirements

Understand throughput

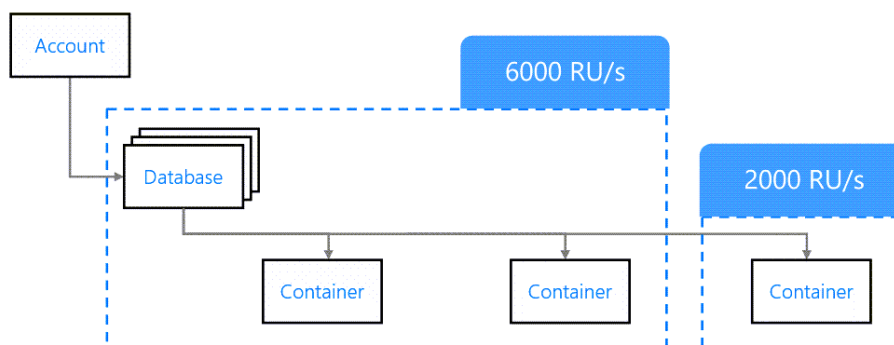
Container-level throughput provisioning



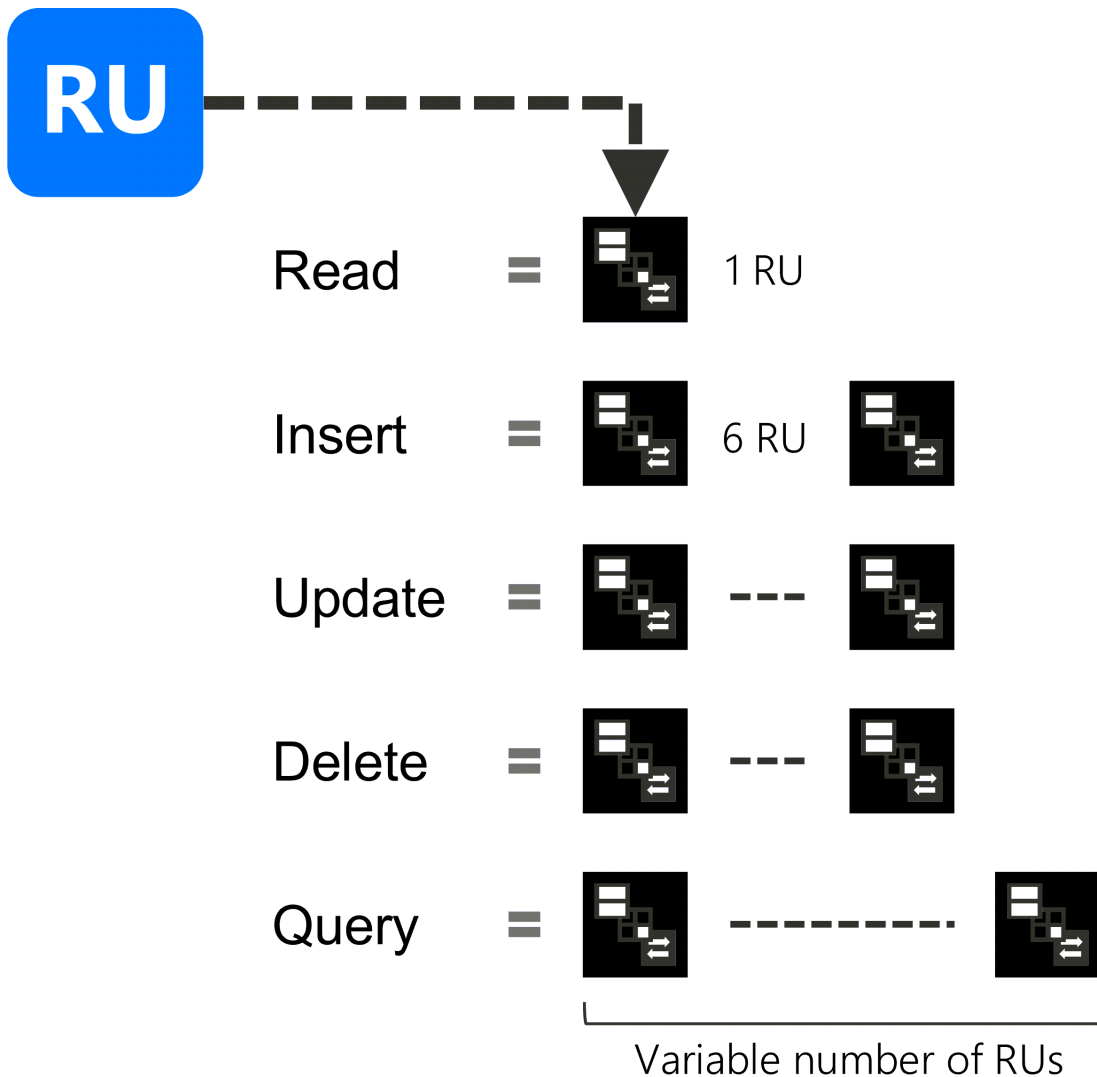
Database-level throughput provisioning



Mixed-throughput provisioning



Evaluate throughput requirements



Operation type	Number of requests per second	Number of RU per request	RU/s needed
Write Single Document	10,000	10	100,000
Top Query #1	700	100	70,000
Top Query #2	200	100	20,000
Top Query #3	100	100	10,000
Total RU/s			200,000 RU/s

Evaluate data storage requirements

Microsoft Azure

Azure Cosmos DB > Capacity Calculator

Sign In

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

Azure Cosmos DB Account Settings

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

API: SQL (Core)

Number of regions: 4

Multi-region writes: ☐ Disabled ☒ Enabled

Workload per region

For a more accurate cost estimate based on your own data, please [sign in](#) and upload your sample data.

Total data stored in transactional store: 1500 GB

Use Analytical Store: ☒ Off ☐ On

Item size: 0 1 KB

Point reads/sec: 5000

Creates/sec: 500

Updates/sec: 250

Deletes/sec:

Cost Estimate

Transactional Storage

Cost per GB/month: USD

Total Data stored per region: x 1,500 GB

EST. STORAGE COST PER MONTH: USD

Transactional Workload

Cost per 100 RU/s per hour with multi-region writes: USD

EST. THROUGHPUT REQUIRED: x 35,250 RU/s

EST. WORKLOAD COST/MONTH: USD

Number of regions: x 4

EST. TOTAL COST/MONTH: USD

[Create Cosmos DB account](#)

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

Time-to-live (TTL)

The TTL value for a container is configured using the `DefaultTimeToLive` property of the container's JSON object.

DefaultTimeToLive	Expiration
<i>Does not exist</i>	Items are not automatically expired
-1	Items will not expire by default
<i>n</i>	<i>n</i> seconds after last modified time

Example:

Container.DefaultTimeToLive	Item.ttl	Expiration in seconds
1000	<i>null</i>	1000
1000	-1	<i>This item will never expire</i>
1000	2000	2000

Another example:

Container.DefaultTimeToLive	Item.ttl	Expiration in seconds
<i>null</i>	<i>null</i>	<i>This item will never expire</i>
<i>null</i>	-1	<i>This item will never expire</i>
<i>null</i>	2000	<i>This item will never expire</i>

Plan for data retention with time-to-live (TTL)

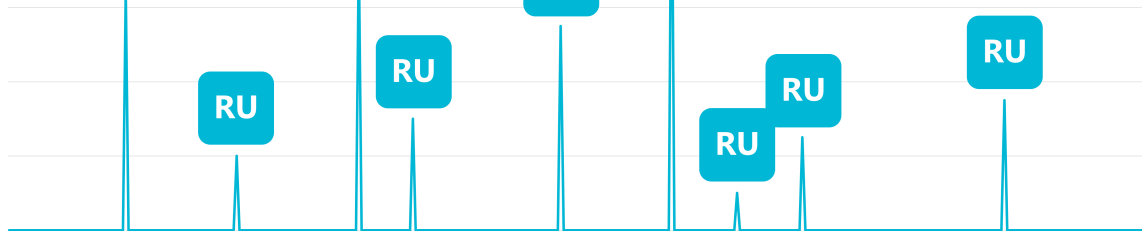
Consider solutions such to aggregate and migrate data such as:

- Change feed
- Azure Data Warehouse
- Azure Blob Storage

Configure Azure Cosmos DB SQL API throughput

Serverless

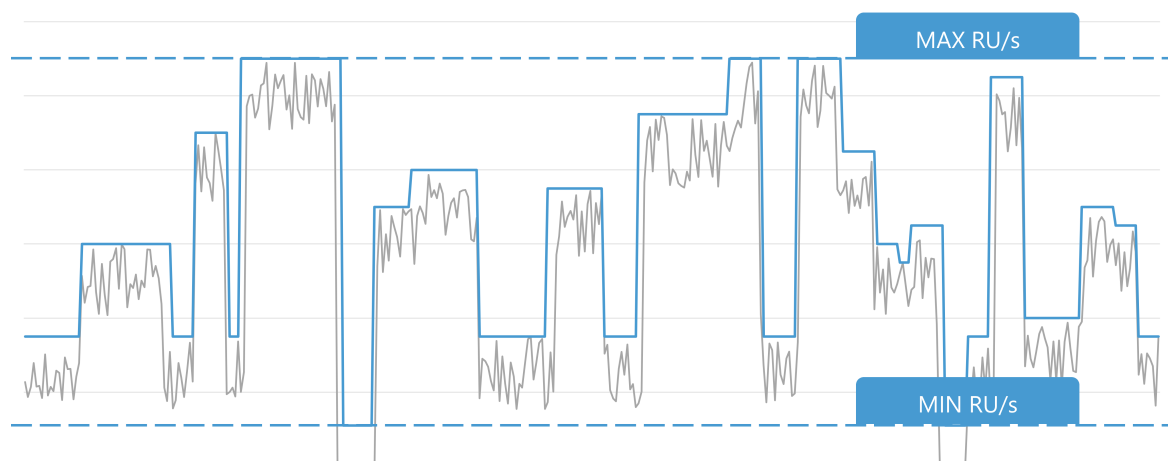




Compare serverless vs. provisioned throughput

Throughput	Workloads	RUs	Global Distribution	Compare storage limits
Provisioned	Ideal for workloads with predictable traffic patterns	Number RUs per second preset to each container	Can distribute data to an unlimited number of Azure regions	Unlimited data in a container
Serverless	Can handle workloads that have wildly varying traffic	Doesn't require any planning or automatic provisioning	Accounts can only run in a single Azure region	Up to 50 GB of data in a container

Autoscale



Compare autoscale vs. standard (manual) throughput

Scaling	Workloads	RUs	Scenarios	Rate-limiting
Standard	Suited for workloads with steady traffic	Requires a static number of request units to be assigned ahead of time	Where the application throughput can be accurately predicted	Since the RU/s are static, requests beyond this will be rate-limited
Autoscale	Suited for unpredictable traffic	You only set the maximum RUs	Where the application throughput can't be accurately predicted, but an acceptable max throughput can be assigned	Will scale up to the max RU/s before similarly rate-limiting responses

Moving data into and out of Azure Cosmos DB SQL API

Move data by using Azure Data Factory

Linked service

```
{
  "name": "<example-name-of-linked-service>",
  "properties": {
    "type": "CosmosDb",
    "typeProperties": {
      "connectionString": "AccountEndpoint=<cosmos-
endpoint>;AccountKey=<cosmos-key>;Database=<cosmos-database>"
    }
  }
}
```

New linked service

Data store Compute

cosmos

All Azure Database File Generic protocol NoSQL Services and apps



Azure Cosmos DB for
MongoDB



Azure Cosmos DB for
NoSQL



Azure Data Lake Storage
Gen1 for Cosmos
Structured Stream



Azure Data Lake Storage
Gen2 for Cosmos
Structured Stream

Continue

Cancel

New linked service (Azure Cosmos DB (SQL API))

Name *

CosmosDb1

Description

Connect via integration runtime * ⓘ

AutoResolveIntegrationRuntime

Authentication method

Connection string

Connection string

Azure Key Vault

Account selection method ⓘ

☒ From Azure subscription ☐ Enter manually

Azure subscription ⓘ

Select all



Azure Cosmos DB account name * ⓘ



Database name *

No database



Additional connection properties

+ New

Annotations

+ New

▸ Parameters

▸ Advanced ⓘ