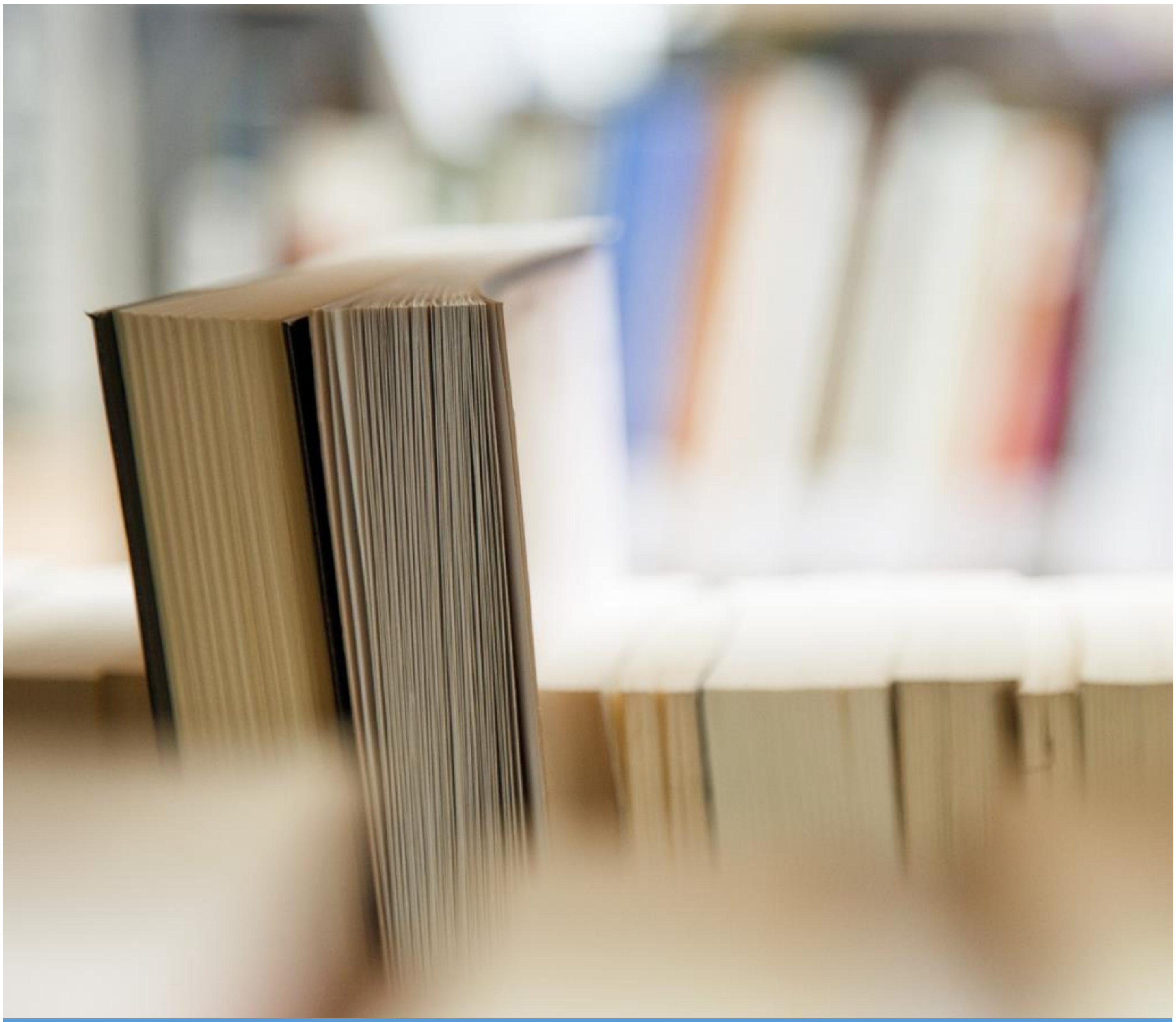


COVID-19 Data Analysis using Azure Cosmos DB

Roy, Sandip (Cognizant) 4/24/20

Table of Contents

1. Introduction

This document outlines how to use Azure Services to perform analysis on COVID-19 data captured from publicly exposed APIs and also build a fundamental dashboard (not focusing too much on the details on the dashboard side) using Power BI to give user end to end experience.

2. Solution Components

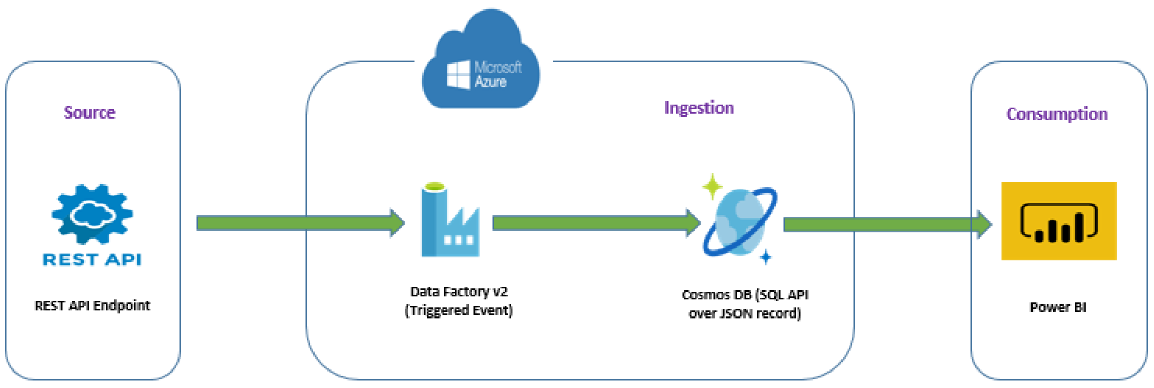
A **RESTful API** is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data and is also referred to as a RESTful web service -- is based on representational state transfer (REST) technology, an architectural style and approach to communications often used in web services development.

Azure Data Factory is a cloud-based data integration service that allows you to create data-driven workflows in the cloud that orchestrate and automate data movement and data transformation. The Data Factory service allows you to create data pipelines that move and transform data and then run the pipelines on a specified schedule.

Azure Cosmos DB is Microsoft's globally distributed, multi-model database service. It enables you to elastically and independently scale throughput and storage across any number of Azure regions worldwide. You can elastically scale throughput and storage, and take advantage of fast, single-digit-millisecond data access using your all possible NoSQL API variants (i.e. columnar, key-value, document and graph) including: SQL, MongoDB, Cassandra, Tables, or Gremlin. Cosmos DB provides comprehensive service level agreements (SLAs) for throughput, latency, availability, and consistency guarantees, something no other database service offers.

Power BI is a business analytics service by Microsoft. It aims to provide interactive visualizations and business intelligence capabilities with an interface simple enough for end users to create their own reports and dashboards.

3. Solution Diagram

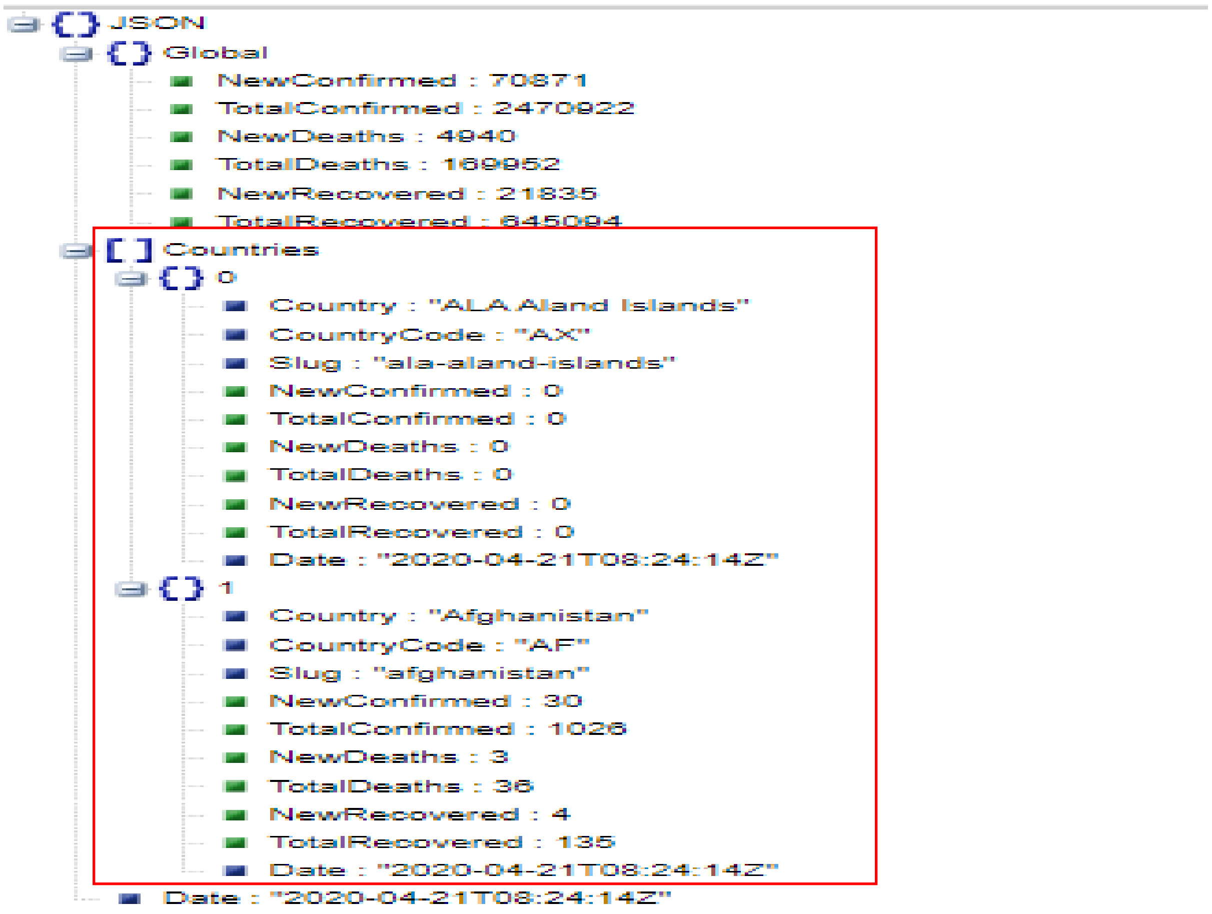


4. Detailed Steps

Step 1 - Identify the REST API you need to consume along with the pre-requisites (credentials, API keys or Auth. tokens etc.) handy.

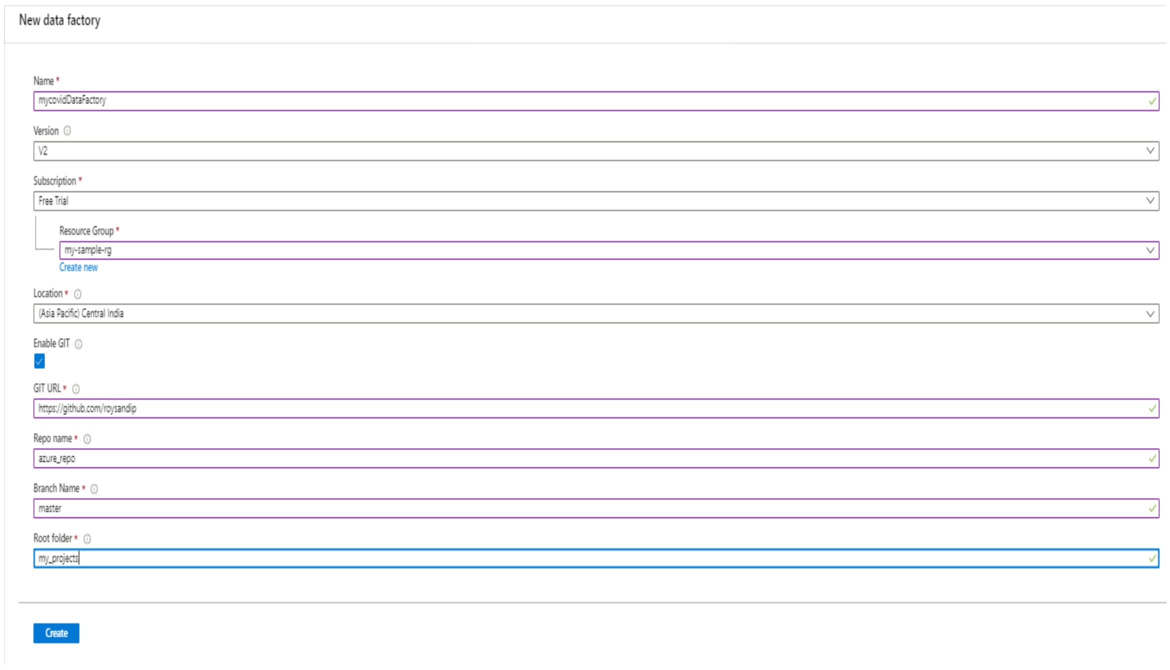
In practical cases, normally the API you would consume would be secured by standard REST API security mechanisms but here for simplicity, let's proceed with simple case (no authentication needed) with anonymous user (refer to **References** section for more details on how secured REST API should be consumed with ADF v2).

We would be using one of public REST API hosted at https://covid19api.com/ for our COVID-19 data analysis where it is updated multiple sides at API end and we are also ingesting data multiple times (not necessarily synced with API refresh but just scheduled as per certain time interval e.g. twice a day or so). The sample COVID-19 JSON data structure looks like following. We would map only highlgihted part for our analysis.



```
{} JSON
  {} Global
      NewConfirmed : 70871
      TotalConfirmed : 2470922
      NewDeaths : 4940
      TotalDeaths : 169952
      NewRecovered : 21835
      TotalRecovered : 645094
  [] Countries
    {} 0
        Country : "ALA Aland Islands"
        CountryCode : "AX"
        Slug : "ala-aland-islands"
        NewConfirmed : 0
        TotalConfirmed : 0
        NewDeaths : 0
        TotalDeaths : 0
        NewRecovered : 0
        TotalRecovered : 0
        Date : "2020-04-21T08:24:14Z"
    {} 1
        Country : "Afghanistan"
        CountryCode : "AF"
        Slug : "afghanistan"
        NewConfirmed : 30
        TotalConfirmed : 1026
        NewDeaths : 3
        TotalDeaths : 36
        NewRecovered : 4
        TotalRecovered : 135
        Date : "2020-04-21T08:24:14Z"
    Date : "2020-04-21T08:24:14Z"
```

Step 2 – Create the ADF pipeline with HTTP as source and Cosmos DB as destination.


New data factory

Name *
mycovidDataFactory

Version
V2

Subscription *
Free Trial

Resource Group *
my-sample-rg
Create new

Location *
(Asia Pacific) Central India

Enable GIT
✓

GIT URL *
https://github.com/roysandip

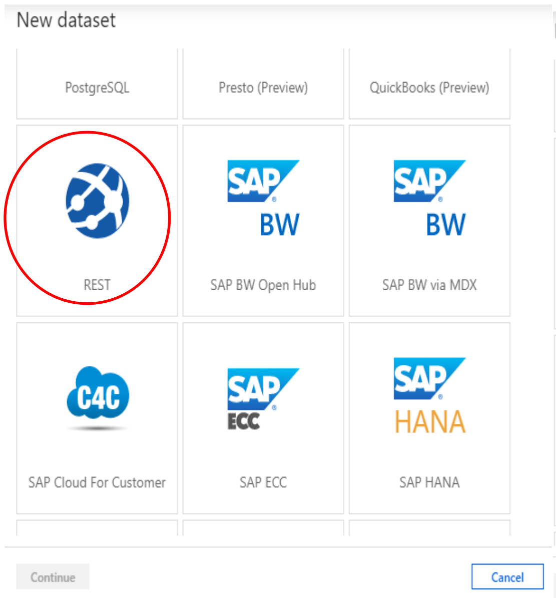
Repo name *
azure_repo

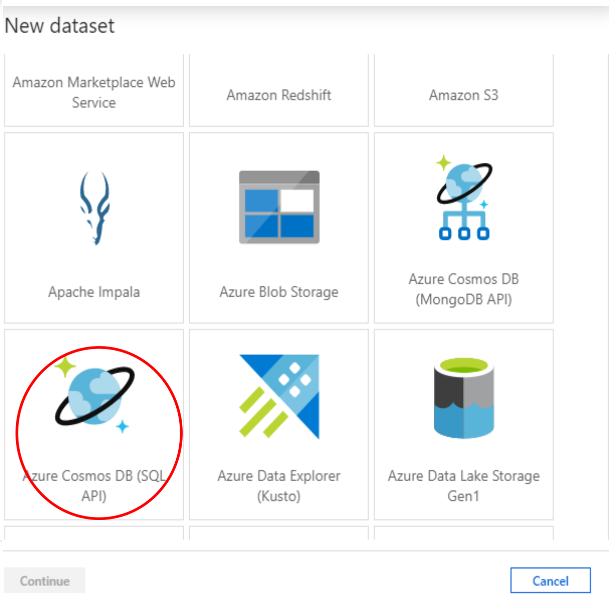
Branch Name *
master

Root folder *
my_projects

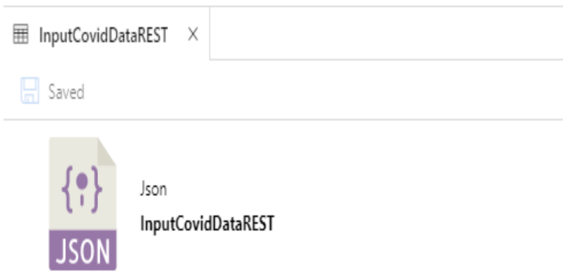
Please note that with ADF v2, you have the option to save/commit your data factory configuration/code directly to GIT (onetime OAuth2 authentication is done against Azure ADF app) repository. You can leverage the same if you want to save it in your desired repository (as I have used my personal GIT repo in this example).

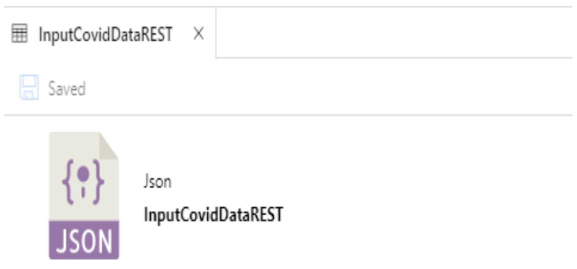
Step 3 – As the Data Factory skeleton is created, let's define the pipeline (source/target/mapping etc.) now. Select source (REST/HTTP) and target (Cosmos DB) from Azure portal as shown below.

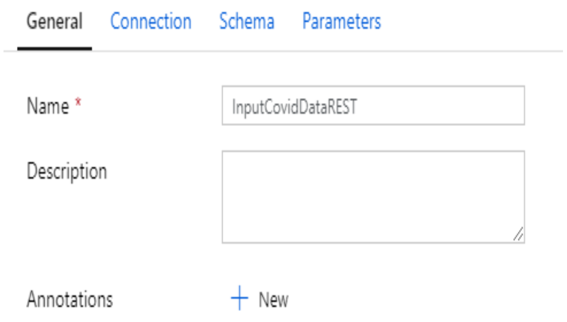



New dataset
Amazon Marketplace Web Service
Amazon Redshift
Amazon S3
Apache Impala
Azure Blob Storage
Azure Cosmos DB (MongoDB API)
zure Cosmos DB (SQL API)
Azure Data Explorer (Kusto)
Azure Data Lake Storage Gen1
Continue
Cancel

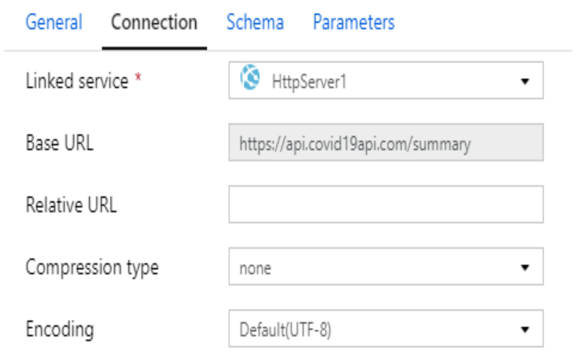
Source





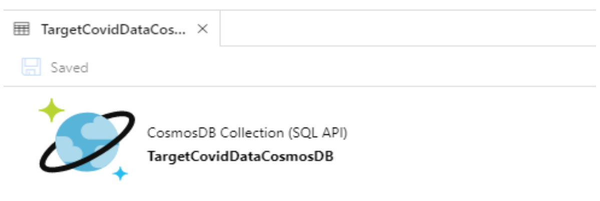


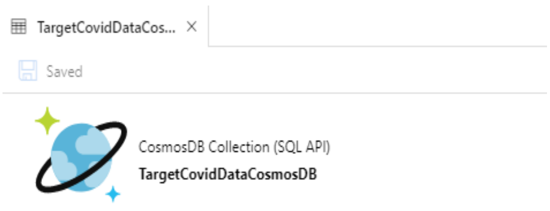
Name *	InputCovidDataREST
Description	

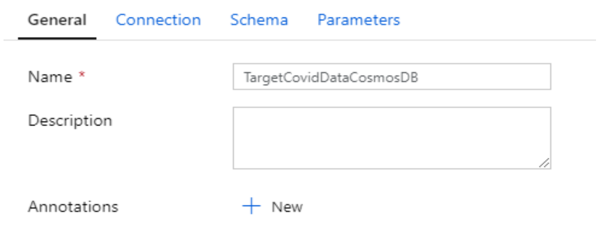


Linked service *	HttpServer1
Base URL	https://api.covid19api.com/summary
Relative URL	
Compression type	none
Encoding	Default(UTF-8)

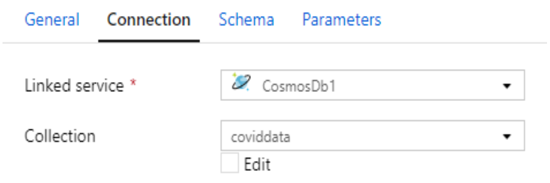
Target



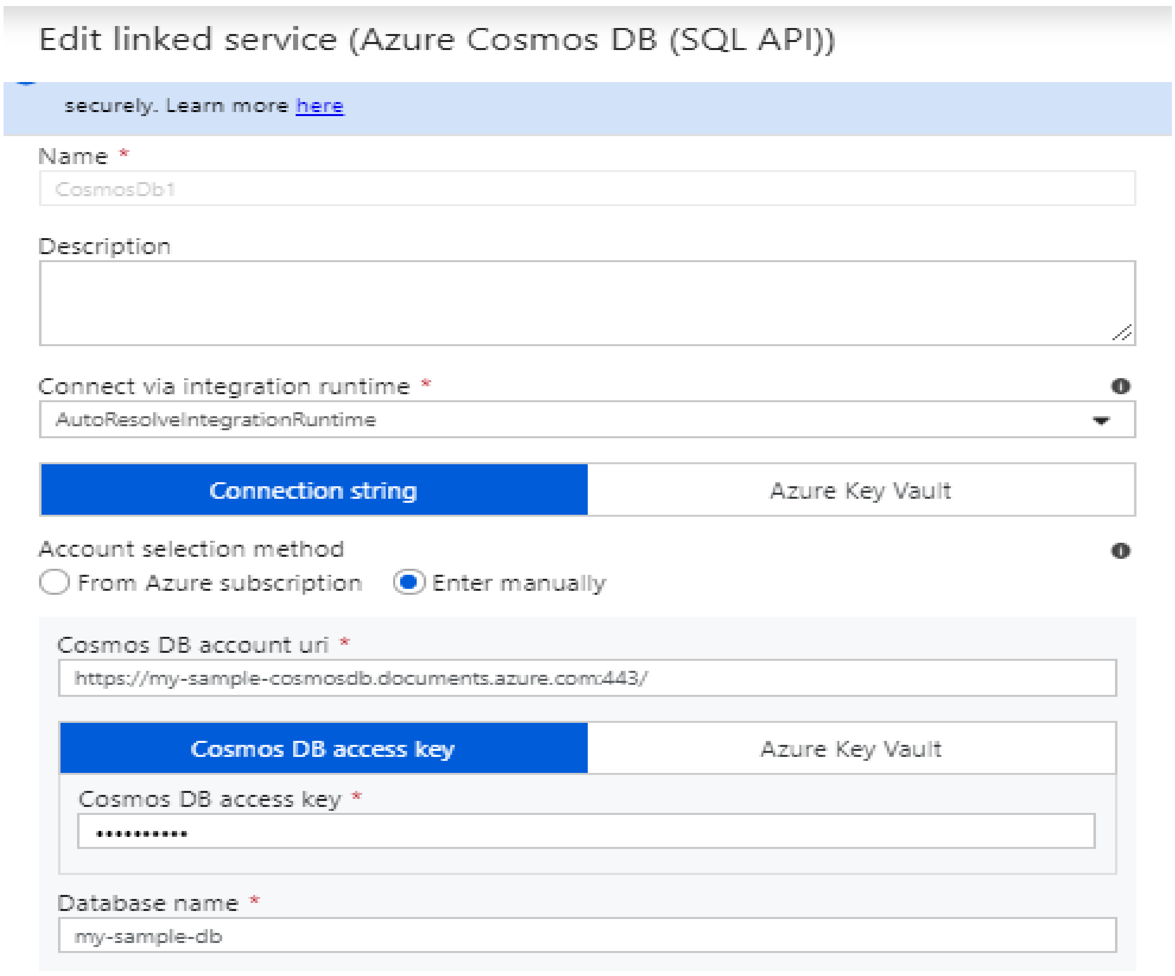




Name *	TargetCovidDataCosmosDB
Description	



Linked service *	CosmosDb1
Collection	coviddata



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

securely. Learn more here

Name *
CosmosDb1

Description

Connect via integration runtime *
AutoResolveIntegrationRuntime

[✓ Connection string] Azure Key Vault

Account selection method
From Azure subscription [✓ Enter manually]

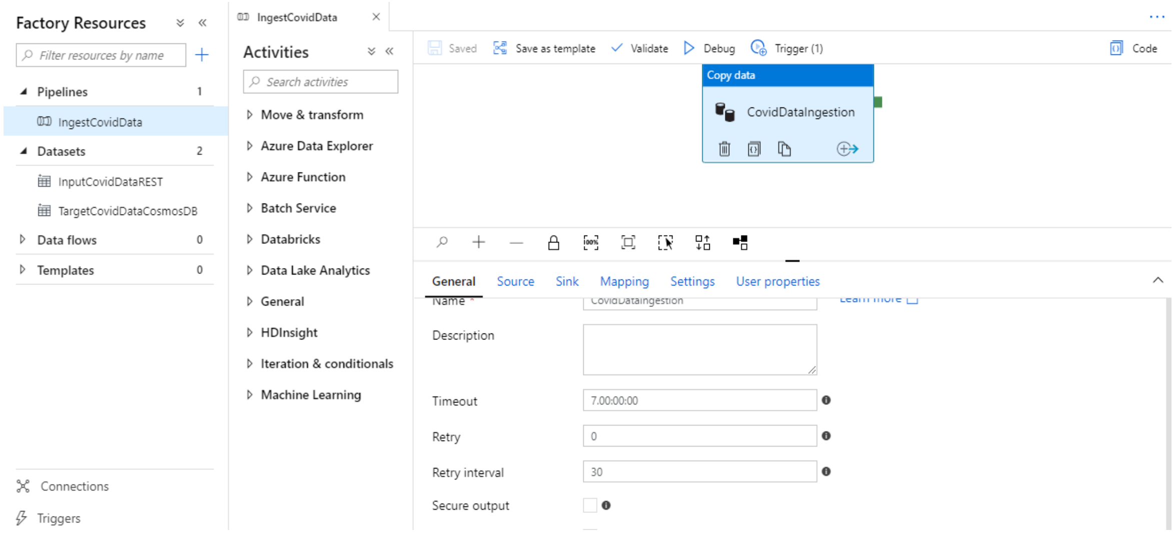
Cosmos DB account uri *
https://my-sample-cosmosdb.documents.azure.com:443/

[✓ Cosmos DB access key] Azure Key Vault

Cosmos DB access key *
•••••••••••

Database name *
my-sample-db

Once the source/target is defined and configured, check at the pipeline level whether it looks complete overall.



Factory Resources
- Pipelines 1
  - IngestCovidData
- Datasets 2
  - InputCovidDataREST
  - TargetCovidDataCosmosDB
- Data flows 0
- Templates 0

Connections
Triggers

IngestCovidData

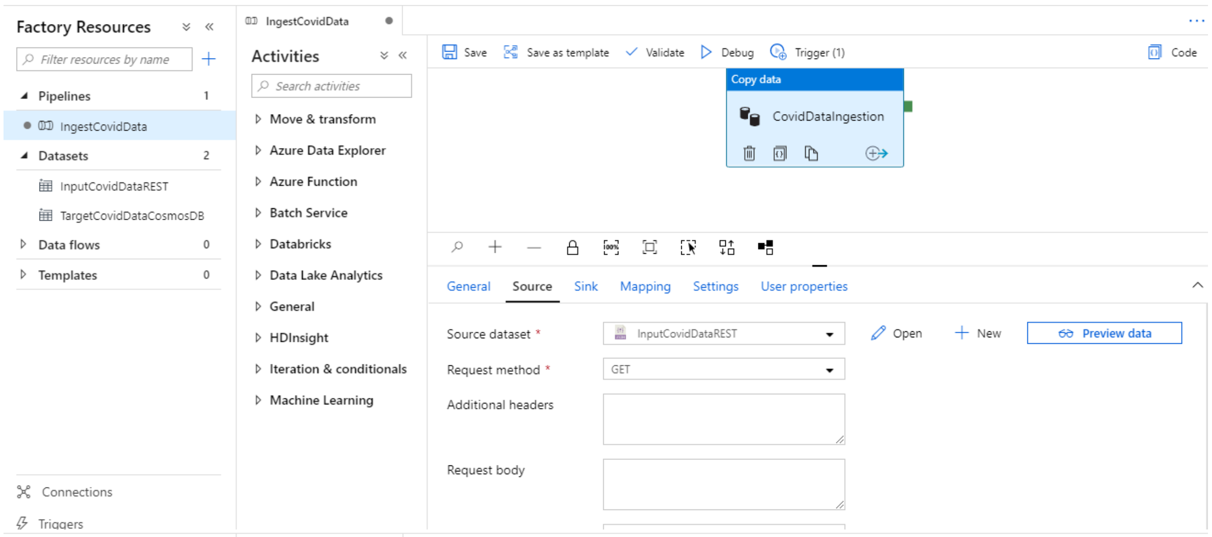
Activities
- Move & transform
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning

Saved | Save as template | Validate | Debug | Trigger (1) | Code

Copy data
CovidDataIngestion

General | Source | Sink | Mapping | Settings | User properties

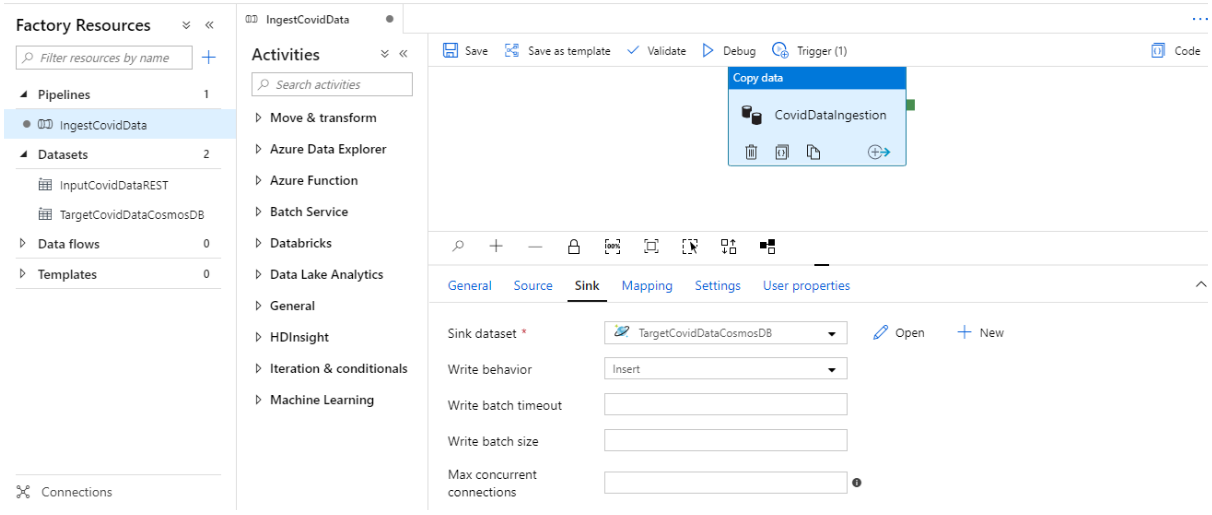
Name * CovidDataIngestion
Description
Timeout 7.00:00:00
Retry 0
Retry interval 30
Secure output


Source dataset: InputCovidDataREST

Request method: GET

Additional headers:

Request body:


Sink dataset: TargetCovidDataCosmosDB

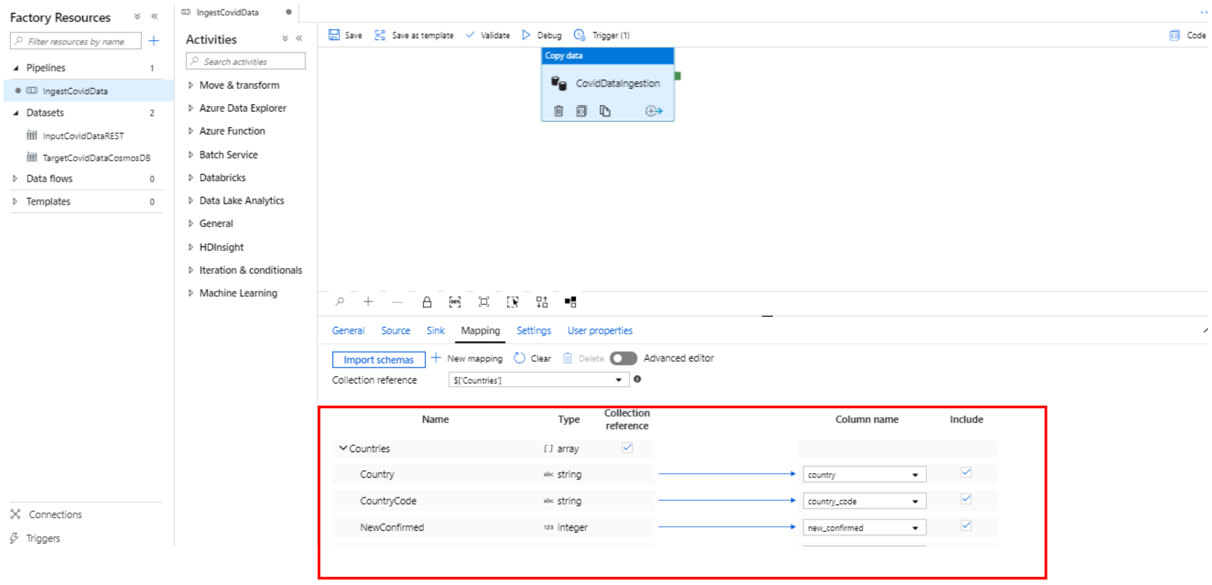
Write behavior: Insert

Write batch timeout:

Write batch size:

Max concurrent connections:

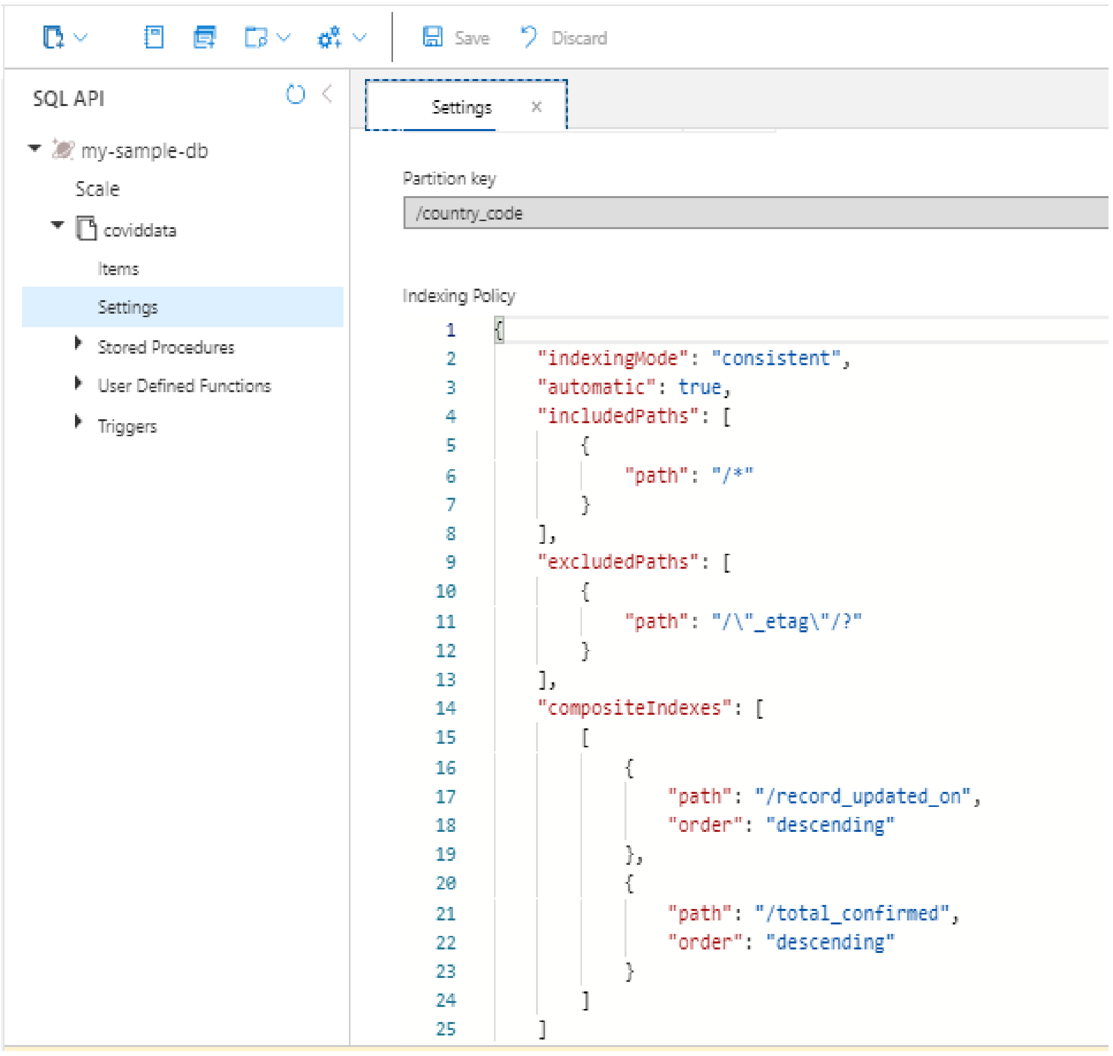
Specify the data mapping between source and target. In our case, as I said before we would be mapping the *Countries* array part only.


Collection reference: $['Countries']

Name	Type	Collection reference	Column name	Include
Countries	array	✓		
Country	string		country	✓
CountryCode	string		country_code	✓
NewConfirmed	integer		new_confirmed	✓

Couple of points while designing Cosmos DB collection (traditionally called table) –

- You need to choose a partition key mandatorily for the records getting stored as the performance/billing cost of executing cross partition query may be significantly higher. In our case, we have chosen *country_code* attribute as partition key meaning each country's data would be maintained in separate partition.
- Composite index must be in place for set of attributes that are to be used together in ORDER BY clause. In our case, we would query the collection using ORDER BY clause with two attributes record_updated_on and total_confirmed to get the most number of affected people along with country name.
- By default, all attributes present in document are indexed, in case we don't want to include certain attribute to be indexed, we can specify that in excludePaths tag.



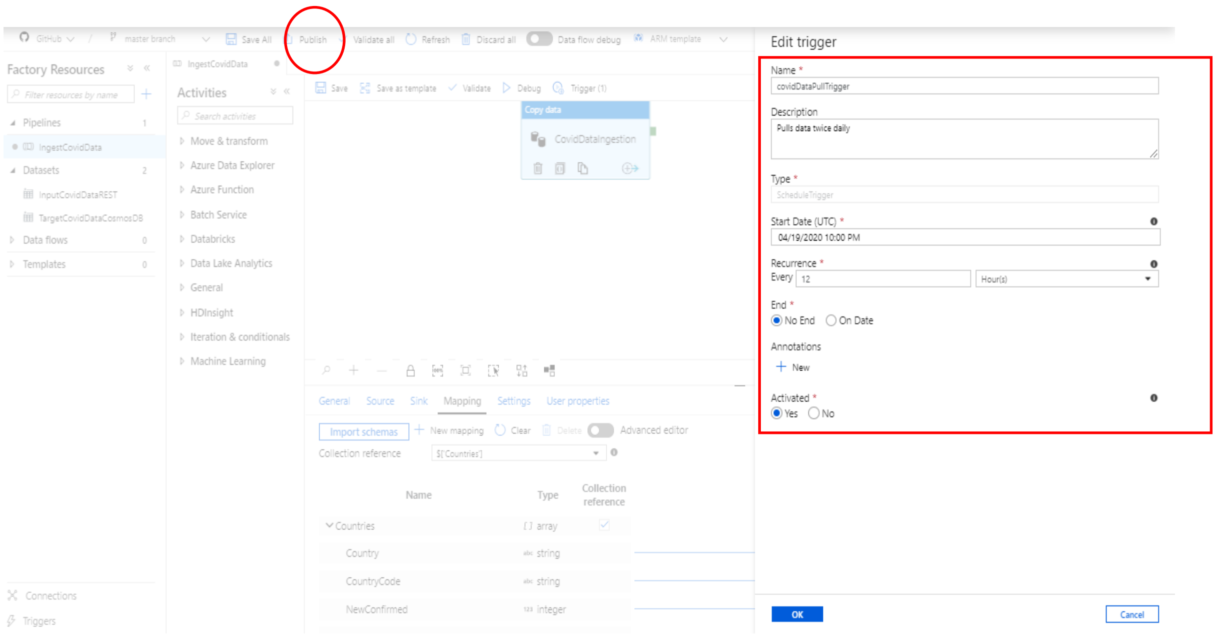
Partition key

/country_code

Indexing Policy

```
{
    "indexingMode": "consistent",
    "automatic": true,
    "includedPaths": [
        {
            "path": "/*"
        }
    ],
    "excludedPaths": [
        {
            "path": "/\"_etag\"/?"
        }
    ],
    "compositeIndexes": [
        [
            {
                "path": "/record_updated_on",
                "order": "descending"
            },
            {
                "path": "/total_confirmed",
                "order": "descending"
            }
        ]
    ]
```

Step 4 - Once the pipeline is completely built, execute it (onetime or through scheduled trigger)


Edit trigger

Name *: covidDataPullTrigger

Description: Pulls data twice daily

Type *: ScheduleTrigger

Start Date (UTC) *: 04/19/2020 10:00 PM

Recurrence *: Every 12 Hour(s)

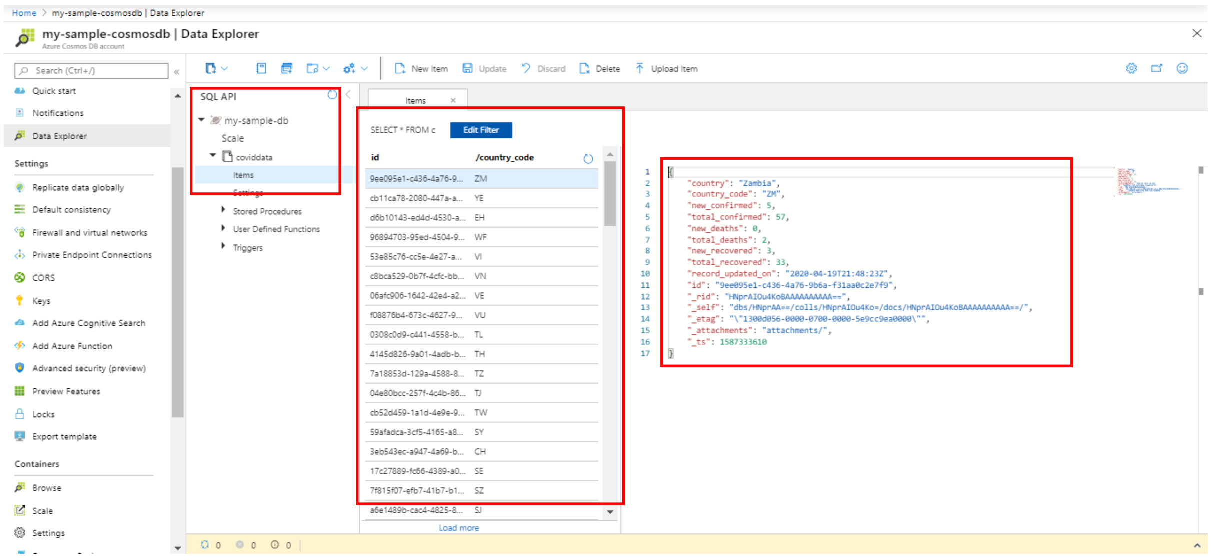
End *: [✓ No End] On Date

Annotations: + New

Activated *: [✓ Yes] No

Name	Type	Collection reference
Countries	[ ] array	✓
Country	string	
CountryCode	string	
NewConfirmed	integer	

Step 5 - After the ADF pipeline is executed successfully, check whether data is loaded in CosmosDB collection or not. It will be stored as document (JSON) only and we will use SQL API to traverse through the document (we can traverse through the nested structure also in case we have same).


my-sample-cosmosdb | Data Explorer

SELECT * FROM c

id	/country_code
9ee095e1-c436-4a76-9...	ZM
cb11ca78-2080-447a-a...	YE
d6b10143-ed4d-4530-a...	EH
96894703-95ed-4504-9...	WF
53e85c76-cc5e-4e27-a...	VI
c8bca529-0b7f-4cfc-bb...	VN
06afc906-1642-42e4-a2...	VE
f08876b4-673c-4627-9...	VU
0308c0d9-c441-4558-b...	TL
4145d826-9a01-4adb-b...	TH
7a18853d-129a-4588-8...	TZ
04e80bcc-257f-4c4b-86...	TJ
cb52d459-1a1d-4e9e-9...	TW
59afadca-3cf5-4165-a8...	SY
3eb543ec-a947-4a69-b...	CH
17c27889-fc66-4389-a0...	SE
7f815f07-efb7-41b7-b1...	SZ
a6e1489b-cac4-4825-8...	SJ

```
{
    "country": "Zambia",
    "country_code": "ZM",
    "new_confirmed": 5,
    "total_confirmed": 57,
    "new_deaths": 0,
    "total_deaths": 2,
    "new_recovered": 3,
    "total_recovered": 33,
    "record_updated_on": "2020-04-19T21:48:23Z",
    "id": "9ee095e1-c436-4a76-9b6a-f31aa0c2e7f9",
    "_rid": "HNprAIOu4KoBAAAAAAAAAA==",
    "_self": "dbs/HNprAA==/colls/HNprAIOu4Ko=/docs/HNprAIOu4KoBAAAAAAAAAA==/",
    "_etag": "\"1300d056-0000-0700-0000-5e9cc9ea0000\"",
    "_attachments": "attachments/",
    "_ts": 1587333610
}
```

You can check your desired queries in the Cosmos DB Data Explorer tab (New Query option). Few sample queries we are considering here for our COVID-19 reporting use case are shown below:

- **India COVID-19 Statistics**

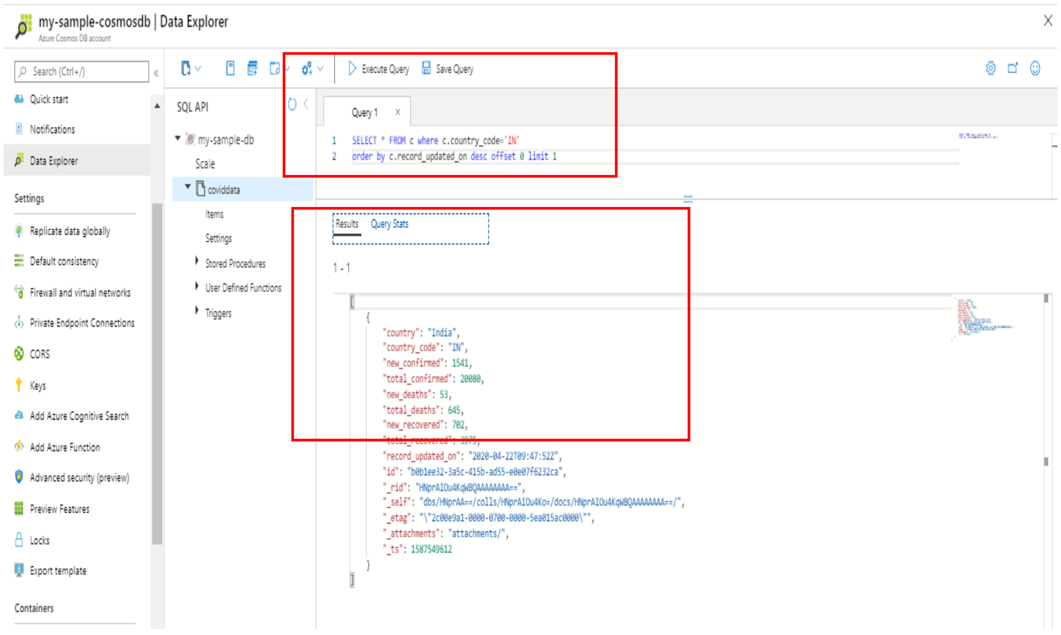
  *SELECT * FROM c where c.country_code='IN' order by c.record_updated_on desc offset 0 limit 1*

- **Top 3 COVID-19 Impacted Countries**

  *SELECT * FROM c order by c.record_updated_on desc,c.total_confirmed desc offset 0 limit 3*

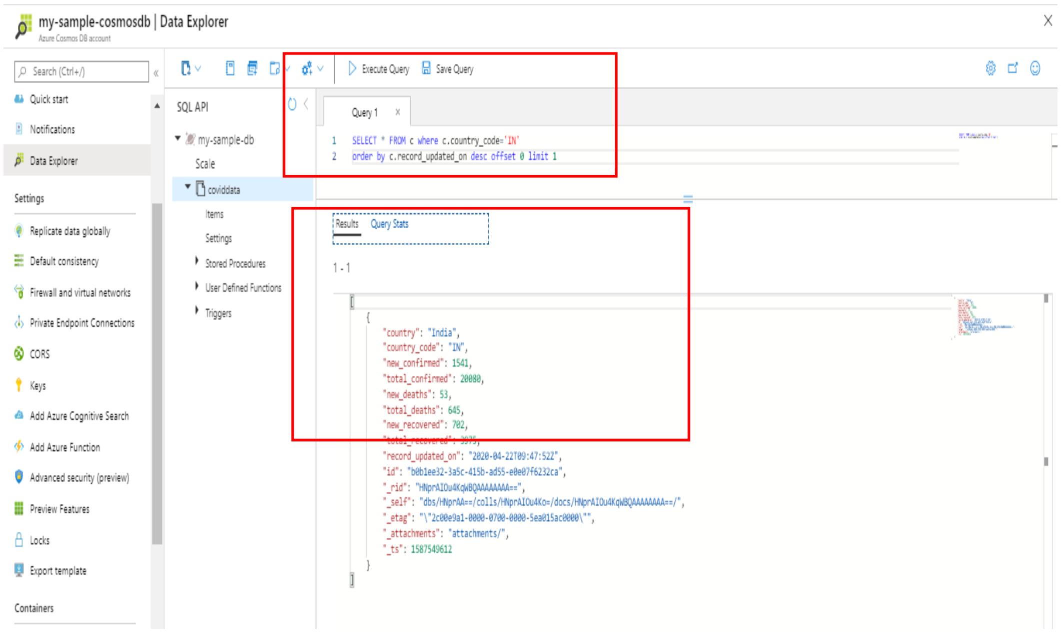
- **COVID-19 Today Worldwide**

  SELECT sum(c.new_confirmed) as new_confirmed,sum(c.total_confirmed) as total_confirmed
  ,sum(c.new_deaths) as new_deaths,sum(c.total_deaths) as total_deaths
  ,sum(c.new_recovered) as new_recovered,sum(c.total_recovered) as total_recovered
  ,c.record_updated_on from c
  where substring(c.record_updated_on,0,10)=substring(GetCurrentDateTime (),0,10)
  group by c.record_updated_on offset 0 limit 1

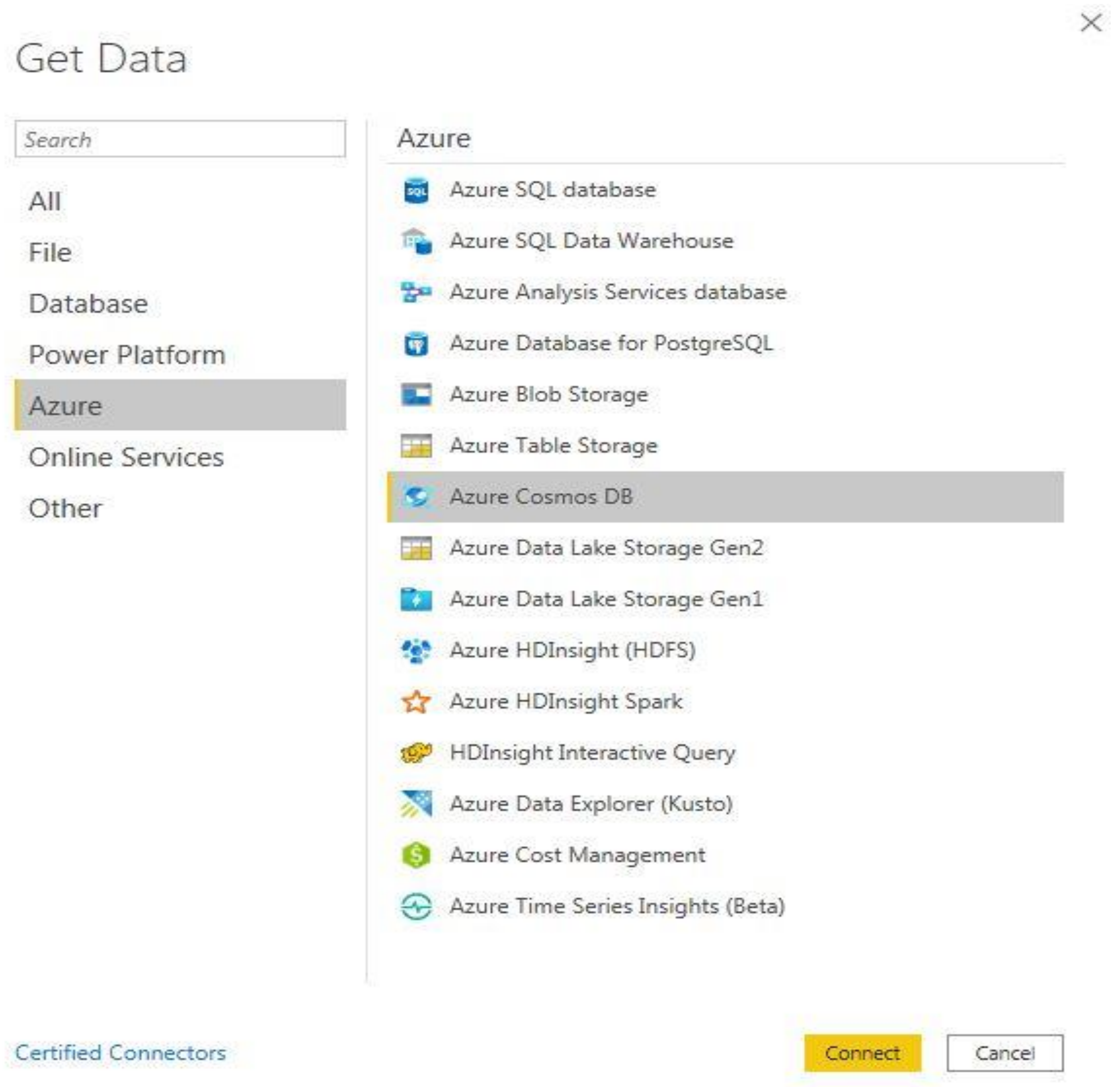


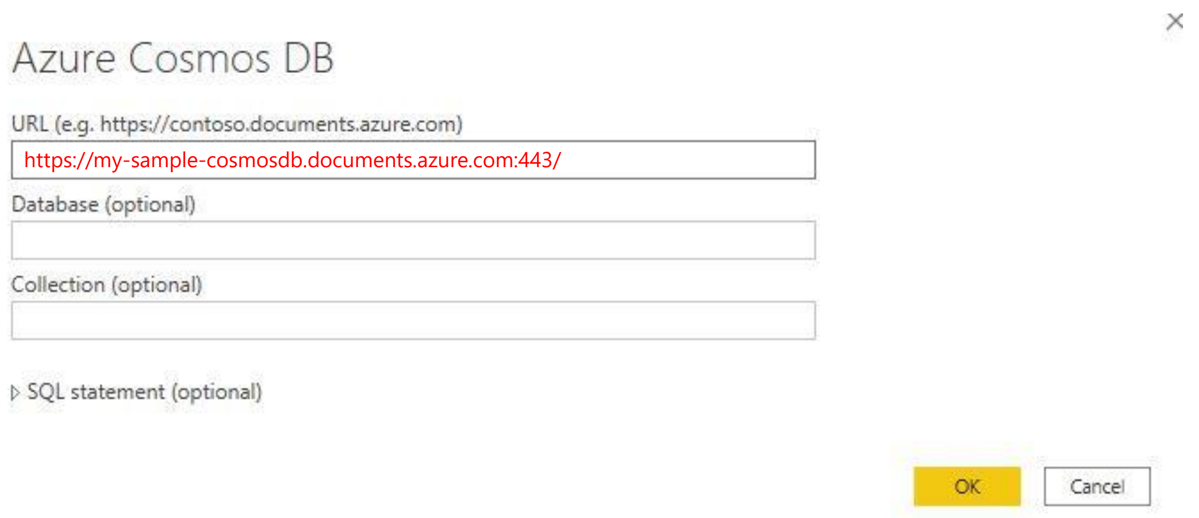
```
SELECT * FROM c where c.country_code='IN'
order by c.record_updated_on desc offset 0 limit 1
```

```
[
    {
        "country": "India",
        "country_code": "IN",
        "new_confirmed": 1541,
        "total_confirmed": 20080,
        "new_deaths": 53,
        "total_deaths": 645,
        "new_recovered": 702,
        "total_recovered": 3975,
        "record_updated_on": "2020-04-22T09:47:52Z",
        "id": "b0b1ee32-3a5c-415b-ad55-e0e07f6232ca",
        "_rid": "HNprAIOu4KqWBQAAAAAAAA==",
        "_self": "dbs/HNprAA==/colls/HNprAIOu4Ko=/docs/HNprAIOu4KqWBQAAAAAAAA==/",
        "_etag": "\"2c00e9a1-0000-0700-0000-5ea015ac0000\"",
        "_attachments": "attachments/",
        "_ts": 1587549612
    }
]
```



Step 6 – Launch Power BI application from Desktop and establish connection with Azure Cosmos DB with connecting URL and Access Key.

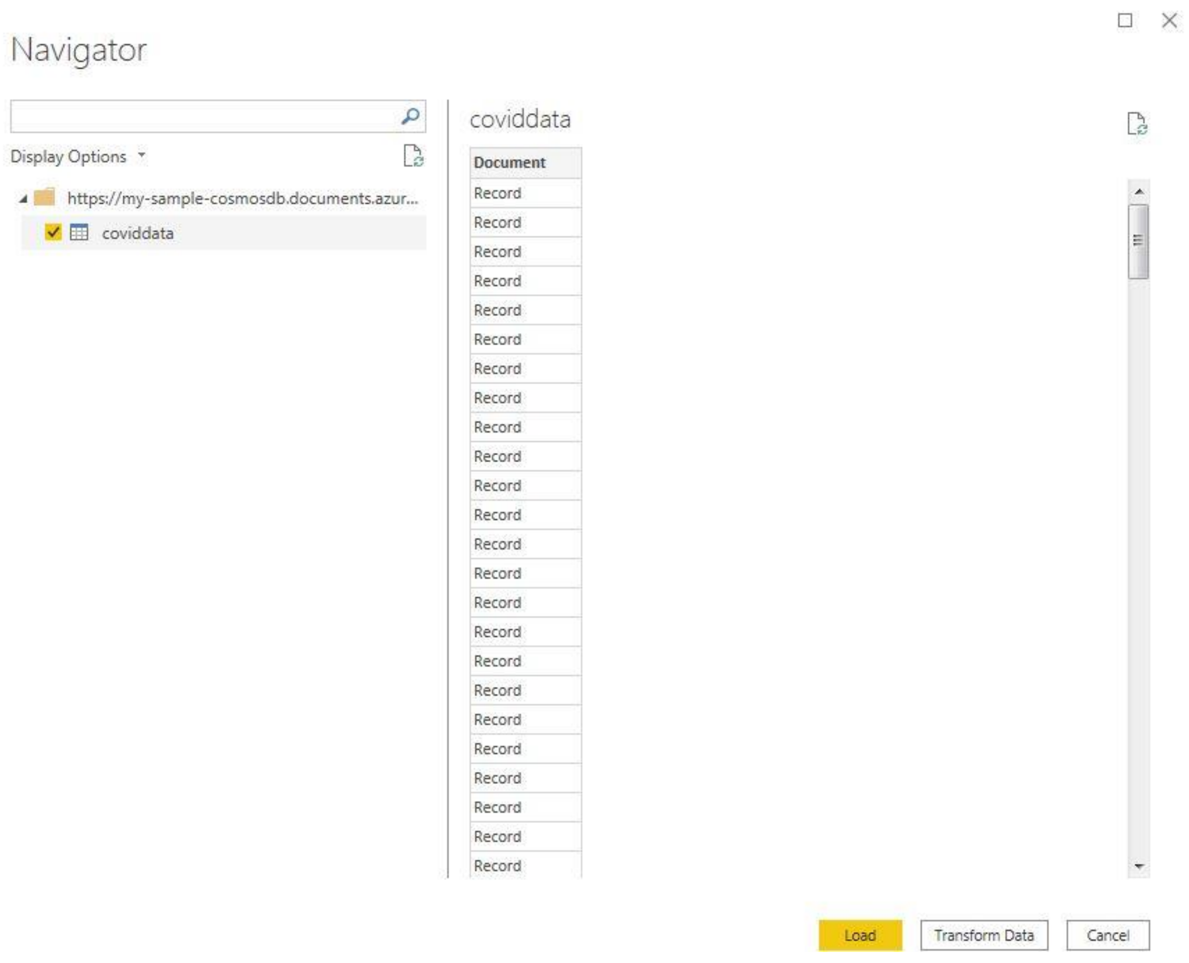




URL (e.g. https://contoso.documents.azure.com)
https://my-sample-cosmosdb.documents.azure.com:443/

Database (optional)

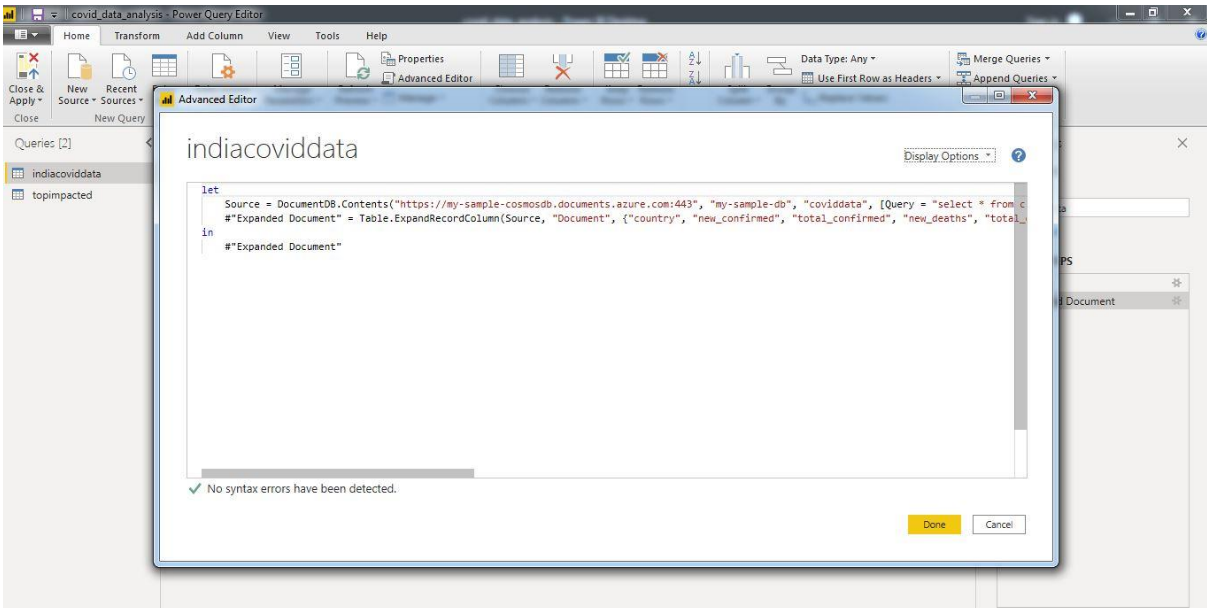
Collection (optional)



coviddata

Document
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record
Record

Please note that once the connection is established, you need to specify the exact Cosmos DB SQL query in Power Editor Window (Advanced Editor) which is supposed to bring/import the required data to Power BI layer and on top of which you can model the required data in Power BI to be used for dashboard/reporting purpose.

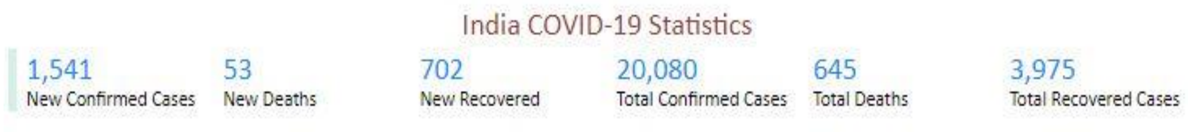


```
let
    Source = DocumentDB.Contents("https://my-sample-cosmosdb.documents.azure.com:443", "my-sample-db", "coviddata", [Query = "select * from c
    #"Expanded Document" = Table.ExpandRecordColumn(Source, "Document", {"country", "new_confirmed", "total_confirmed", "new_deaths", "total_
in
    #"Expanded Document"
```

No syntax errors have been detected.

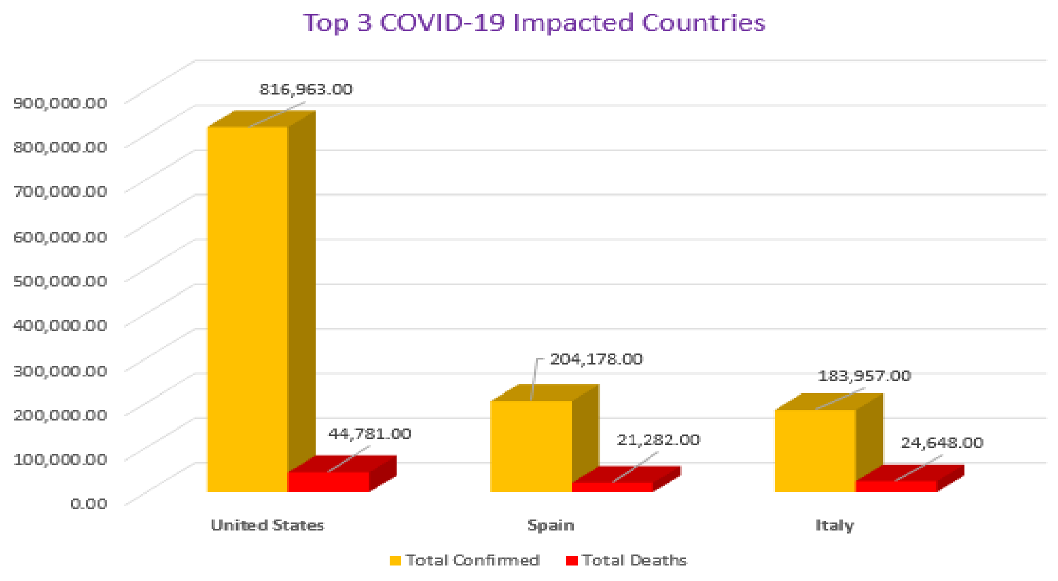
In this example, we have used periodic refresh feature from Power BI to refresh the reporting data (imported from Cosmos DB through query defined in PowerEditor (as shown above) but in case of near real time data refresh for this use case, we would need Spark in between Cosmos DB and Power BI as Power BI do not support DirectQuery for Cosmos DB but it has a support for Spark (refer to **References** section below for more details on how to integrate HDI Spark in this pipeline)

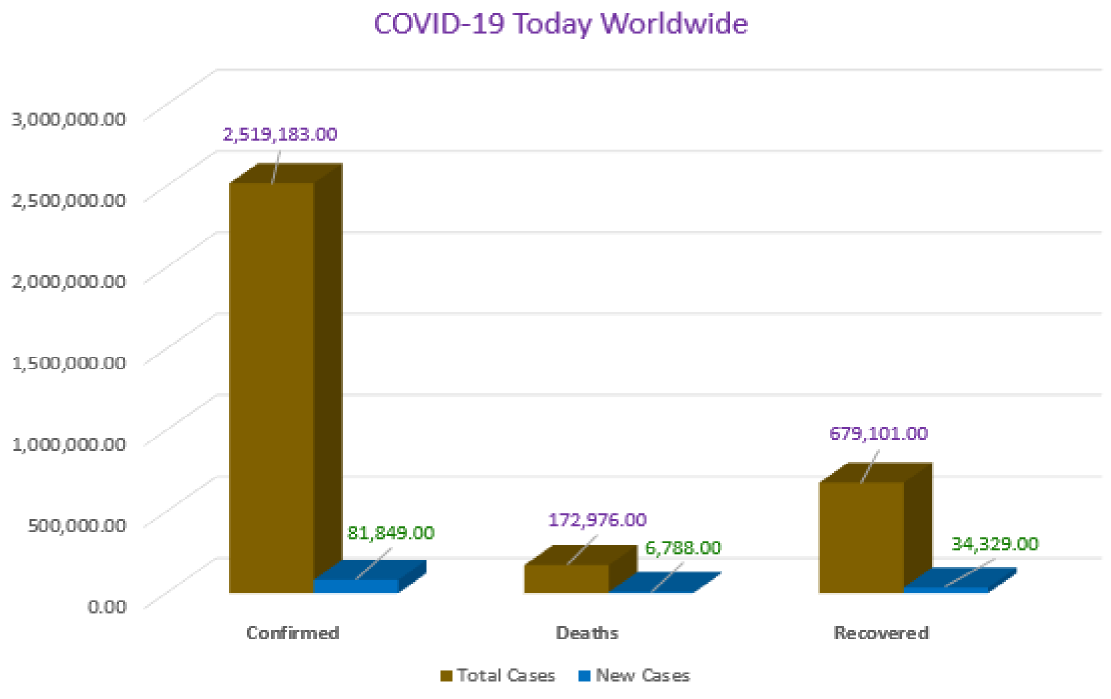
<u>Sample Dashboards</u> (based on 3 scenarios we mentioned above):



India COVID-19 Statistics

1,541	53	702	20,080	645	3,975
New Confirmed Cases	New Deaths	New Recovered	Total Confirmed Cases	Total Deaths	Total Recovered Cases





5. References

https://www.alexvolok.com/2019/adfv2-rest-api-part1-oauth2/
https://github.com/Azure/azure-cosmosdb-spark/wiki/Configuring-Power-BI-Direct-Query-to-Azure-Cosmos-DB-via-Apache-Spark-(HDI)
https://docs.microsoft.com/en-us/azure/cosmos-db/introduction