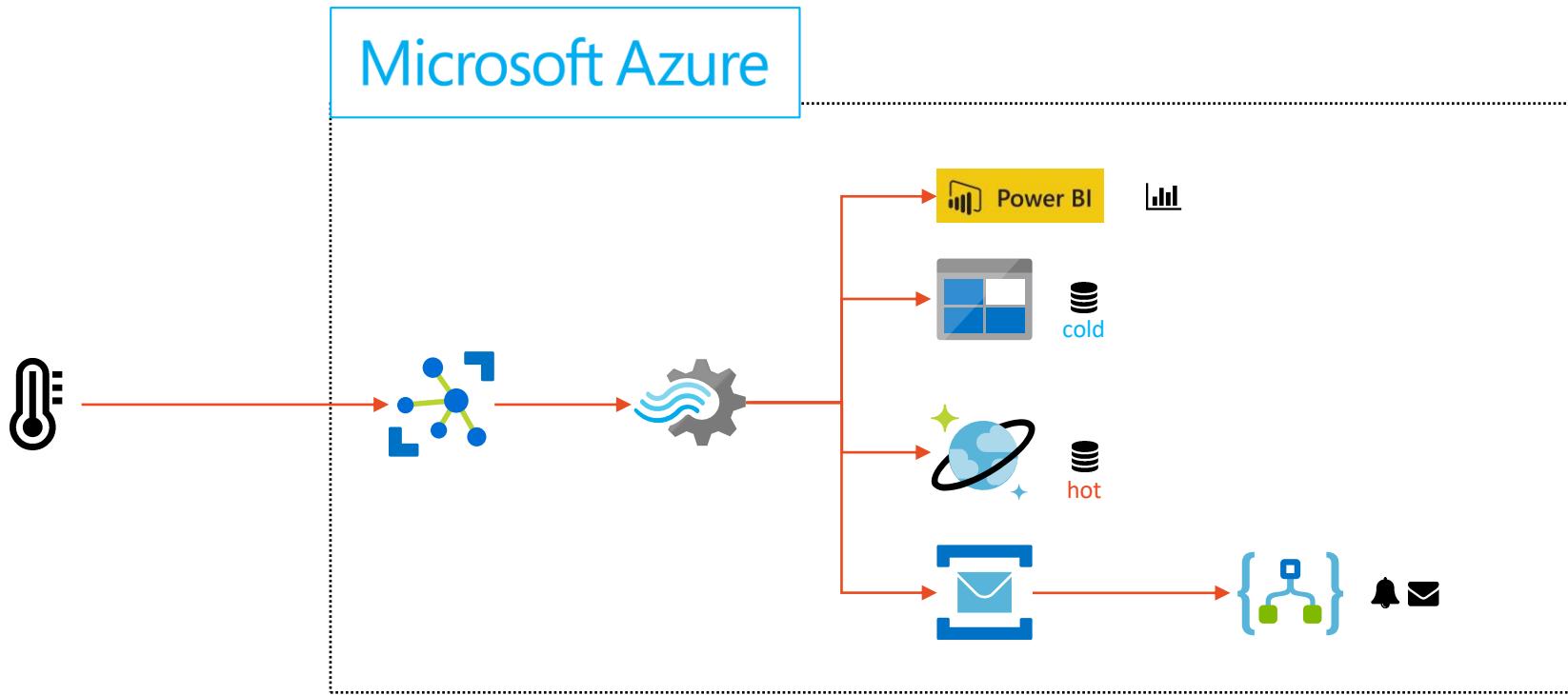




Microsoft  
Azure



# Telemetry Stream Analytics



# Build and deploy solution

## Microsoft Azure IoT SDK for Node.js



This repository contains the following SDKs:

- Azure IoT Hub Device SDK: to connect devices to Azure IoT Hub. [API Reference](#)
- Azure IoT Hub Service SDK: enables developing back-end applications making use of Azure IoT Hub. [API Reference](#)
- Azure IoT Hub Provisioning Device SDK: to connect devices to the Azure IoT Hub Provisioning Service. [API Reference](#)
- Azure IoT Hub Provisioning Service SDK: enables developing back-end applications making use of the Azure IoT Provisioning Service. [API Reference](#)

## Developing applications for Azure IoT

Visit [Azure IoT Dev Center](#) to learn more about developing applications for Azure IoT.

## How to use the Azure IoT SDKs for Node.js

Devices and data sources in an IoT solution can range from a simple network-connected sensor to a powerful, standalone computing device. Devices may have limited processing capability, memory, communication bandwidth, and communication protocol support. The IoT device SDKs enable you to implement client applications for a wide variety of devices.

The SDK team publishes the SDKs as [npm](#) packages:

- Azure IoT Hub Device SDK
  - [Device Client](#) npm package 1.9.9
  - [MQTT Transport](#) npm package 1.9.9
  - [AMQP Transport](#) npm package 1.9.9
  - [HTTP Transport](#) npm package 1.9.9

[azure-iot-sdk-node](#)

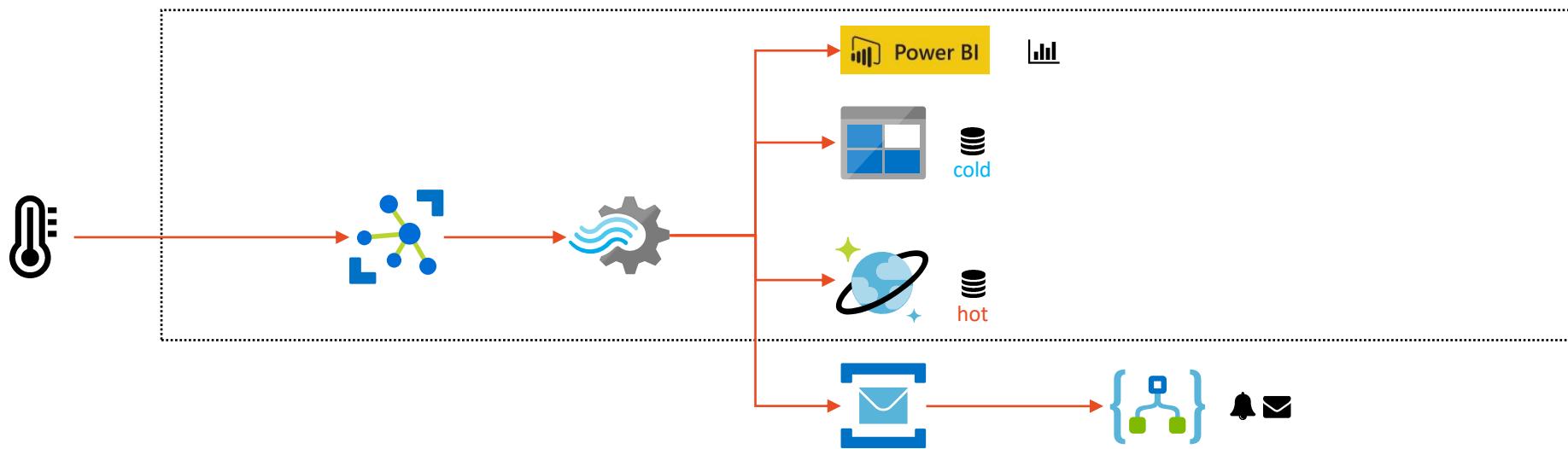
For the purpose of the article, we will simulate a device through the [Microsoft Azure device SDK](#) for Node.js. Our example will simulate a simple telemetry device, with the following structure:

```
{  
    "id": "spaceneedle"  
    ,temperature: 20 + (Math.random() * 10)  
    ,localisation: {  
        type: "Point"  
        , coordinates: [2.3488, 48.8534]  
    }  
    ,Owner: {  
        CompanyName: "Miscrosoft"  
        ,id_companyName: "4309718283412"  
    }  
}
```

# Create a Power BI account

[https://signup.microsoft.com/signup?Sku=a403ebcc-fae0-4ca2-8c8c-7a907fd6c235&ru=https%3A%2F%25app.powerbi.com&pbi\\_source=ASA](https://signup.microsoft.com/signup?Sku=a403ebcc-fae0-4ca2-8c8c-7a907fd6c235&ru=https%3A%2F%25app.powerbi.com&pbi_source=ASA)

# real-time reporting



# Azure IoT Hub creation

step 1

- Access to Azure portal : <https://portal.azure.com/>
- Launch the service “iot hub”

The screenshot shows the Azure portal interface. On the left, there is a sidebar with various service categories like Home, Dashboard, All services, Favorites, etc. A search bar at the top has 'IoT Hub' typed into it. Below the search bar, the results are displayed under the 'Everything' category. The 'IoT Hub' result is highlighted with a red box. To the right of the search results, there are sections for 'Azure IoT Hub Security', 'Notification Hub Namespaces', and 'Notification Hubs'. The main content area shows the 'IoT Hub' blade. It includes a navigation bar with 'Home > IoT Hub', a 'Répertoire par défaut' section, and a table header for 'IoT Hub' with columns: NAME, TYPE, RESOURCE GROUP, LOCATION, and SUBSCRIPTION. A message 'No results.' is shown below the table. At the bottom, there is a 'Create IoT Hub' button.

# Azure IoT Hub creation

step 1

- Create a new ressource group, which will be using for all the exercice

The screenshot shows the 'IoT hub' creation interface on the Microsoft Azure portal. The 'Basics' tab is selected. A modal dialog box is open, prompting for a new resource group name. The dialog contains the following text:  
A resource group is a container that holds related resources for an Azure solution.  
Name: big-data-classes-telemetry-stream-analytics  
Buttons: OK, Cancel.

At the bottom of the main screen, there are three buttons: 'Review + create', 'Next: Size and scale >', and 'Automation options'.

- Review + create

# Azure IoT Hub creation

step 1

Create

IoT hub  
Microsoft

Basics   Size and scale   Review + create

**BASICS**

|                  |   |
|------------------|---|
| Subscription ⓘ   | Essai gratuit                               |
| Resource Group ⓘ | big-data-classes-telemetry-stream-analytics |
| Region ⓘ         | France Central                              |
| IoT Hub Name ⓘ   | bgc-iot-telemetry-stream-analytics          |

**SIZE AND SCALE**

|                              |           |
|------------------------------|-----------|
| Pricing and scale tier ⓘ     | S1        |
| Number of S1 IoT Hub units ⓘ | 1         |
| Messages per day ⓘ           | 400 000   |
| Cost per month               | 21.08 EUR |

[Create](#)   [« Previous: Size and scale](#)   [Automation options](#)

# Azure IoT Hub device creation

## step 2

- Create a new device

The screenshot shows two side-by-side Azure IoT Hub management pages.

**Left Panel (IoT Hub):** This panel lists various management options for the IoT Hub. The 'IoT devices' option is highlighted in blue, indicating it is the active section. Other visible options include 'Query explorer', 'Automatic Device Management' (with 'IoT Edge' and 'IoT device configuration'), 'Messaging' (with 'File upload' and 'Message routing'), 'Resiliency' (with 'Manual failover (preview)'), 'Security' (with 'Overview', 'Security Alerts', 'Recommendations', 'Resources', and 'Custom Alerts'), and 'Monitoring' (with 'Alerts').

**Right Panel (bgc-iot-telemetry-stream-analytics - IoT devices):** This panel displays a list of IoT devices. At the top, there are buttons for '+ Add', 'Refresh', and 'Delete'. A search bar is present above the device list. The list itself is currently empty, showing the message: 'You can use this tool to view, create, update, and delete devices on your IoT Hub.' Below the list, there is a query builder with fields for 'Field' (set to 'Select or enter your own'), 'Operator' (set to '='), and 'Value'. A button '+ Add new clause' is available to build more complex queries. At the bottom, there is a table header with columns: DEVICE ID, STATUS, LAST ACTIVITY, LAST STATUS UPDA..., AUTHENTICATION ..., CLOUD TO DEVICE ...

# Azure IoT Hub device creation

## step 2

Create a device

Find Certified for Azure IoT devices in the Device Catalog

\* Device ID ⓘ spaceneedle ✓

Authentication type ⓘ

Symmetric key X.509 Self-Signed X.509 CA Signed

\* Primary key ⓘ Enter your primary key

\* Secondary key ⓘ Enter your secondary key

Auto-generate keys ⓘ

Connect this device to an IoT hub ⓘ

Enable Disable

Parent device ⓘ

No parent device Set a parent device

Save

In «IoT Devices » pane:

- ID: enter a name keep it in mind for the other steps
- Authentication type: Symmetric Key
- Auto Generate keys: checked
- Connect device to IoT Hub : enable

# Azure IoT Hub configuration

## step 3

In your new created device, copy the “Connection string (primary key)”

The screenshot shows the 'Device details' page for a device named 'spaceneedle'. The top navigation bar includes 'Save', 'Message to device', 'Direct method', 'Device twin', 'Add module identity', 'Regenerate keys', and 'Refresh' buttons. The main configuration area contains fields for 'Device Id' (spaceneedle), 'Primary key' (redacted), 'Secondary key' (redacted), 'Connection string (primary key)' (redacted), and 'Connection string (secondary key)' (redacted). Below these fields are buttons for enabling or disabling device connection to an IoT hub ('Enable' is selected) and a section for setting a 'Parent device' (labeled 'No parent device'). A 'Module identities' tab is active, showing a note about associated module identities and a table with columns: 'MODULE IDENTITY NAME', 'CONNECTION STATE', 'CONNECTION STATE LAST UPDATED', and 'LAST ACTIVITY TIME'. The table displays the message 'No module identities listed.'

# Link solution to Azure IoT Hub

## step 4

In your sdk code, paste the “Connection string (primary key)”

```
// Copyright (c) Microsoft. All Rights Reserved.
// Licensed under the MIT license. See LICENSE file in the project root for full license information.

'use strict';

var Protocol = require('azure-iot-device-mqtt').Mqtt;
// Uncomment one of these transports and then change it in fromConnectionString to test other transports
// var Protocol = require('azure-iot-device-amqp').AmqpWs;
// var Protocol = require('azure-iot-device-http').Http;
// var Protocol = require('azure-iot-device-amqp').Amqp;
// var Protocol = require('azure-iot-device-mqtt').MqttWs;
var Client = require('azure-iot-device').Client;
var Message = require('azure-iot-device').Message;

// String containing Hostname, Device Id & Device Key in the following formats:
// "HostName=<iothub_host_name>;DeviceId=<device_id>;SharedAccessKey=<device_key>"
//var connectionString = "HostName=sqlbits-iothub-changefeed.azure-devices.net;DeviceId=devicetwo;SharedAccessKey=tYQr3V3l4C3fZdNL";
var connectionString = 'HostName=bgc-telemetry-stream-analytics.azure-devices.net;DeviceId=spaceneedle;SharedAccessKey=+4/eHWHdbOk';

// fromConnectionString must specify a transport constructor, coming from any transport package.
var client = Client.fromConnectionString(connectionString, Protocol);

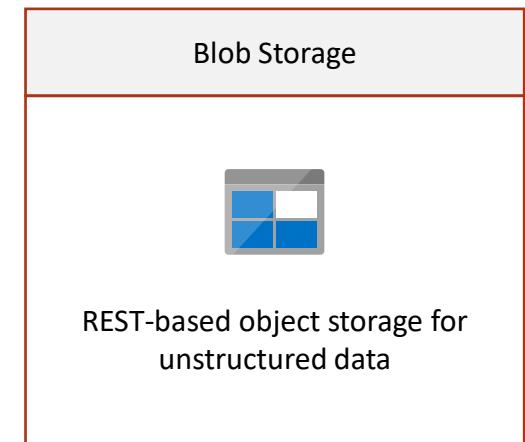
var connectCallback = function (err) {
    if (err) {
        console.error('Could not connect: ' + err.message);
    } else {
        console.log('Client connected');
        client.on('message', function (msg) {
            console.log('Id: ' + msg.messageId + ' Body: ' + msg.data);
            // When using MQTT the following line is a no-op.
            client.complete(msg, printResultFor('completed'));
            // The AMQP and HTTP transports also have the notion of completing, rejecting or abandoning the message.
            // When completing a message, the service that sent the C2D message is notified that the message has been processed.
            // When rejecting a message, the service that sent the C2D message is notified that the message won't be processed by the device.
            // When abandoning the message, IoT Hub will immediately try to resend it. The method to use is client.abandon(msg, callback).
            // MQTT is simpler: it accepts the message by default, and doesn't support rejecting or abandoning a message.
        });
    }
};

// Create a message and send it to the IoT Hub every second
//client.createMessage('Hello IoT Hub').send(function (err) {
```

# Blob Storage Creation

step 5

The screenshot shows the Azure portal interface. At the top, there's a search bar with the text 'blob'. Below it, the 'Storage accounts' service is highlighted in the 'Services' section. The 'Marketplace' section lists several items related to storage, including 'Storage account', 'Microsoft Azure Blob Storage to Alation', 'Flexify.IO - Azure Blob/Amazon S3 Data Migration', and 'Azure Blob Storage on IoT Edge'. The 'Documentation' section contains links to quickstarts and introductions. The main content area shows the 'Storage accounts' blade with a message 'No results were found.' and a note 'Searching all subscriptions. Change'.



# Blob Storage Creation

## step 5

Create storage account

Basics Advanced Tags Review + create

Azure Storage is a Microsoft-managed service providing cloud storage that is highly available, secure, durable, scalable, and redundant. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cost of your storage account depends on the usage and the options you choose below. [Learn more](#)

PROJECT DETAILS

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

\* Subscription: Essai gratuit

\* Resource group: big-data-classes-telemetry-stream-analytics

Create new

INSTANCE DETAILS

The default deployment model is Resource Manager, which supports the latest Azure features. You may choose to deploy using the classic deployment model instead. [Choose classic deployment model](#)

\* Storage account name: telemetryblobstream

\* Location: (Europe) West Europe

Performance: Standard (radio button selected)

Account kind: StorageV2 (general purpose v2)

Replication: Read-access geo-redundant storage (RA-GRS)

Access tier (default): Cool (radio button)

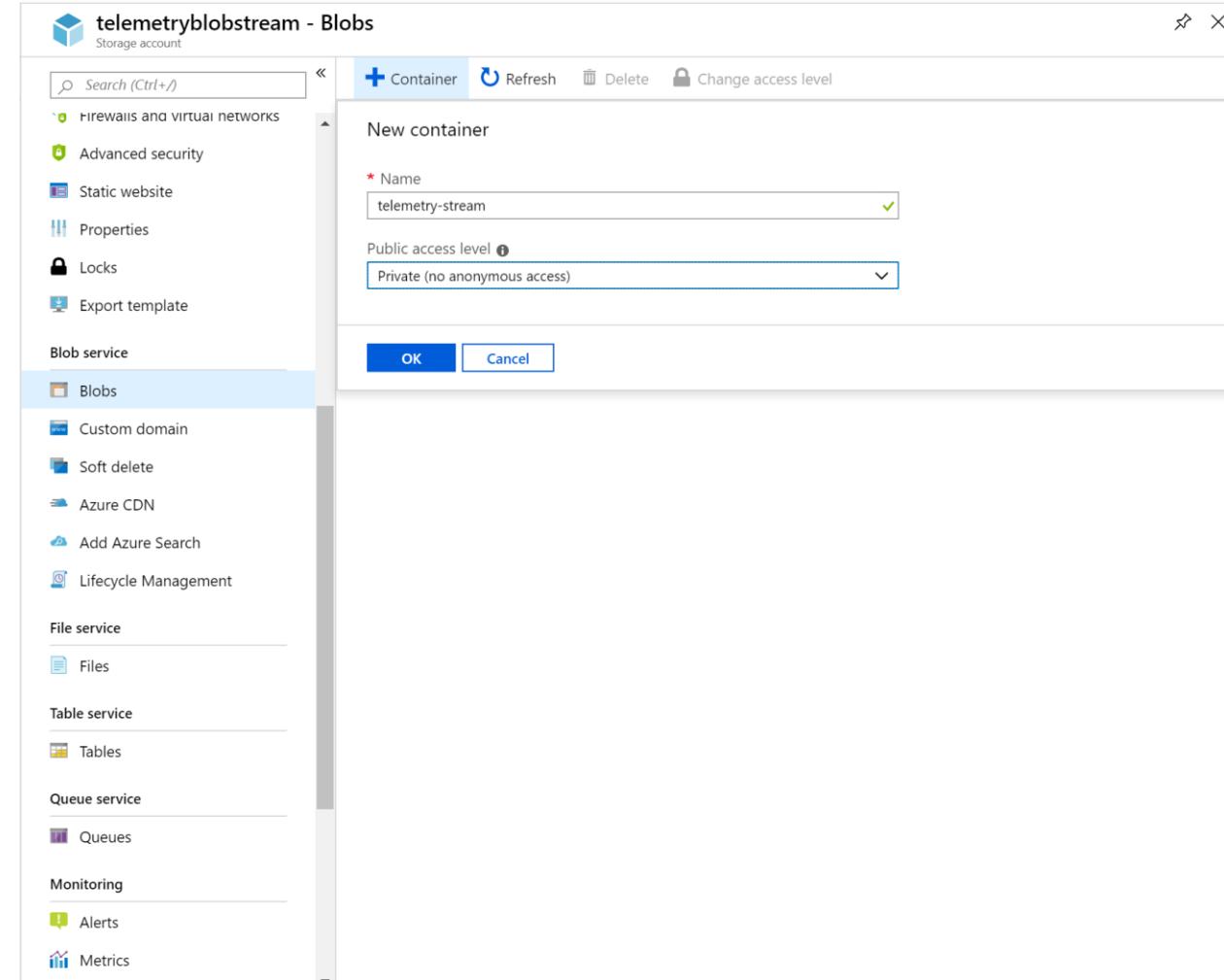
Review + create

Previous

Next : Advanced >

# Create a blob storage container

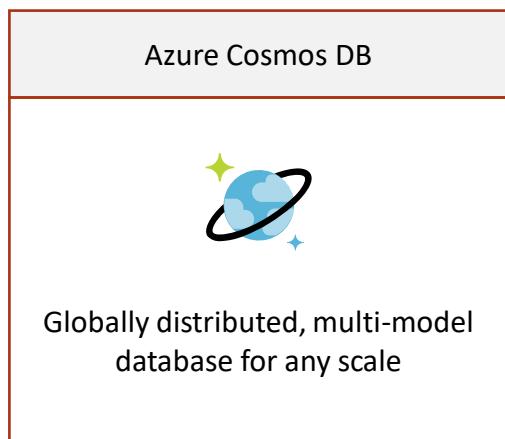
step 6



# Cosmos DB Creation

step 7

The screenshot shows the Azure portal interface. A search bar at the top contains the text "cosmos". The main content area displays the "Azure Cosmos DB" service, specifically the "Subscriptions" section. It shows one item, "bgccosmos", which is identified as an "Azure Cosmos DB account". To the right, there are three vertical tabs: "Marketplace", "Documentation", and "Resource Groups". Under "Marketplace", there are links to "Azure Cosmos DB", "Azure Cosmos DB Reserved Capacity", and "Striim for Real-Time Data Integration to CosmosDB". Under "Documentation", there are links to "Introduction to Azure Cosmos DB | Microsoft Docs", "Azure Cosmos DB Documentation - Tutorials, API Reference ...", "SQL queries for Azure Cosmos DB | Microsoft Docs", and "Develop locally with the Azure Cosmos Emulator | Microsoft ...". Under "Resource Groups", there is a link to "big-data-classes-cosmosdb-analytics".



# Cosmos DB Creation

step 7

Create Azure Cosmos DB Account

Try Cosmos DB for free, up to 20K RU/s, for 30 days with unlimited renewals. [→](#)

Basics Network Tags Review + create

Azure Cosmos DB is a globally distributed, multi-model, fully managed database service. [Try it for free](#), for 30 days with unlimited renewals. Go to production starting at \$24/month per database, multiple containers included. [Learn more](#)

**PROJECT DETAILS**

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

\* Subscription: Essai gratuit

\* Resource Group: big-data-classes-telemetry-stream-analytics

**INSTANCE DETAILS**

\* Account Name: bg-cosmos-stream-analytics

\* API: Core (SQL)

Apache Spark: Enable

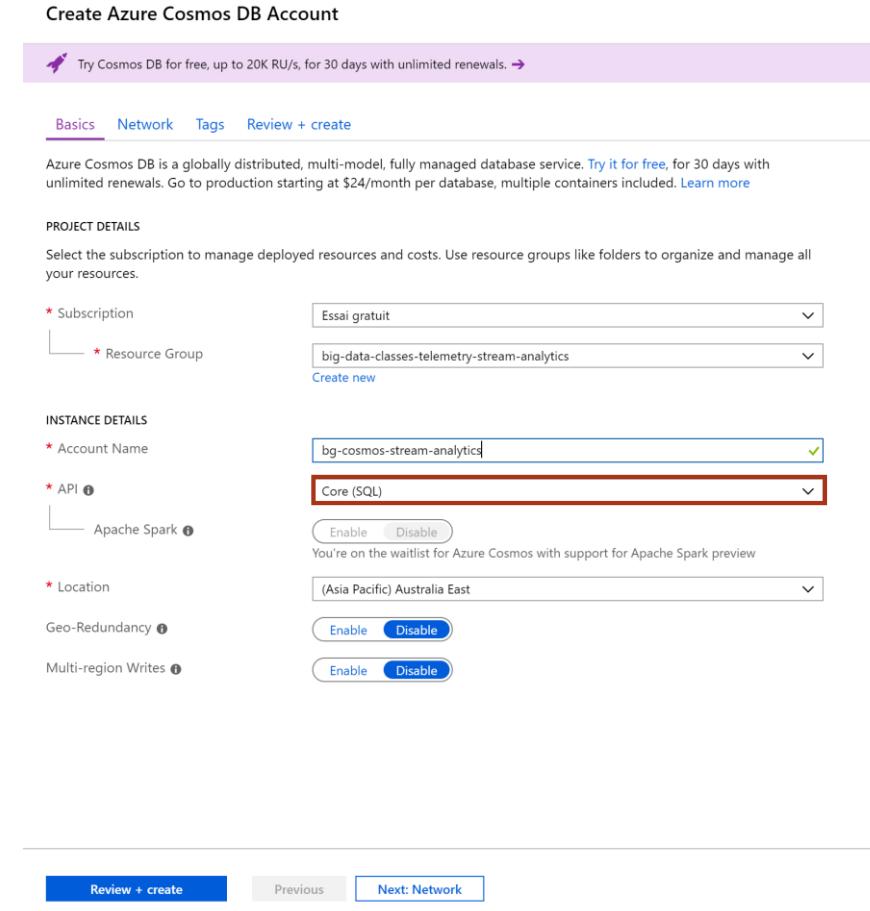
You're on the waitlist for Azure Cosmos with support for Apache Spark preview

\* Location: (Asia Pacific) Australia East

Geo-Redundancy: Enable

Multi-region Writes: Enable

Review + create Previous Next: Network



For the purpose of this architecture we will choose the Core(SQL) for the API.

# Cosmos DB Creation

## step 7

The screenshot shows the Azure Cosmos DB account overview page for 'bg-cosmos-stream-analytics'. The top navigation bar includes options like '+ Add Container', 'Refresh', 'Move', 'Delete Account', 'Data Explorer', and 'Enable geo-redundancy'. The main content area displays account details such as Status (Online), Resource group (big-data-classes-telemetry-stream-analytics), Subscription (Essai gratuit), and URI (https://bg-cosmos-stream-analytics.documents.azure.com:443/). It also shows 'Containers' (none yet) and 'Regions' (a world map showing regions like West Europe and West US). The 'Monitoring' section shows request statistics for the last 24 hours, with a chart indicating 'No data to display'.

# Stream Analytics job creation step 8

All services  X

Everything Stream Analytics jobs ★

General

Stream Analytics job Microsoft

Azure Stream Analytics is a fully managed, cost effective real-time event processing engine that helps to unlock deep insights from data. Stream Analytics makes it easy to set up real-time analytic computations on data streaming from devices, sensors, web sites, social media, applications, infrastructure systems, and more.

With a few clicks in the Azure portal, you can author a Stream Analytics job specifying the input source of the streaming data, the output sink for the results of your job, and a data transformation expressed in a SQL-like language. You can monitor and adjust the scale/speed of your job in the Azure portal to scale from a few kilobytes to a gigabyte or more of events processed per second.

Stream Analytics leverages years of Microsoft Research work in developing highly tuned streaming engines for time-sensitive processing, as well as language integrations for intuitive specifications of such.

[Twitter](#) [Facebook](#) [LinkedIn](#) [YouTube](#) [Google+](#) [Email](#)

---

```
graph LR; IoTHub[IoT Hub] --> SA[Stream Analytics]; SA --> Dashboard[Dashboard]; SA --> Alerts[Alerts]; SA --> Storage[Storage]
```

Create

# Stream Analytics job creation

## step 8

### New Stream Analytics job □ ×

\* Job name  
bgc-streamAnalytics-telemetry-stream-an... ✓

\* Subscription  
Essai gratuit

\* Resource group  
big-data-classes-telemetry-stream-analytics ✓  
[Create new](#)

\* Location  
West Europe

Hosting environment  ⓘ  
Cloud Edge

Streaming units (1 to 192)  ⓘ  
 3

- Name: choose a different name
- Subscription : choose between available subscriptions
- Resource group: select the previously created group
- Hosting environment: cloud

[Create](#)

[Automation options](#)

# Stream Analytics job configuration step 8

The screenshot shows the Azure Stream Analytics job configuration interface. The job name is 'bgc-streamAnalytics-telemetry-stream-analytics'. The left sidebar includes sections for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Locks, Job topology, Inputs, Functions, Query, Outputs), Configure (Storage account settings, Scale), and a preview section. The main area displays the following information:

- Resource group (change) :** big-data-classes-telemetry-stream-analytics
- Status :** ---
- Location :** West Europe
- Subscription (change) :** Essai gratuit
- Subscription ID :** 7ba392bc-90ba-4f4d-a5ab-3c9e0cc8263c
- Send feedback :** UserVoice
- Created :** ---
- Started :** ---
- Output watermark :** ---
- Hosting environment :** Cloud

The 'Inputs' section shows 0 inputs, labeled 'Empty'. The 'Outputs' section shows 0 outputs, also labeled 'Empty'. In the 'Query' section, the following T-SQL code is displayed:

```
1 SELECT
2 *
3 INTO [YourOutputAlias]
4 FROM [YourInputAlias]
```

[Edit query](#)

# Stream Analytics job configuration step 9

input : iot hub

bgc-streamAnalytics-telemetry-stream-analytics - Inputs

Add stream input Add reference input

| NAME  | SOURCE TYPE |
|-------|-------------|
| Empty |             |

Tags  
Diagnose and solve problems

Settings  
Locks

Job topology  
Inputs (selected)  
Functions  
Query  
Outputs

Configure  
Storage account settings  
Scale  
Locale  
Event ordering  
Error policy  
Compatibility level  
Managed Identity

General  
Tools  
Properties

Monitoring  
Metrics

IoT Hub  
New input

\* Input alias  
iot-telemetry-stream

Provide IoT Hub settings manually  
 Select IoT Hub from your subscriptions

Subscription  
Essai gratuit

IoT Hub  
bgc-iot-telemetry-stream-analytics

Endpoint  
Messaging

Shared access policy name  
iothubowner

Shared access policy key  
\*\*\*\*\*

Consumer group  
\$Default

\* Event serialization format  
JSON

Encoding  
UTF-8

Event compression type  
None

Save

# Stream Analytics job configuration step 9

output : Blob storage

netry-stream-analytics - Outputs

bgc-streamAnalytics-telemetry-stream-analytics - Outputs

Stream Analytics job

Search (Ctrl+ /)

Add

NAME

Empty

Tags

Diagnose and solve problems

Settings

Locks

Job topology

Inputs

Functions

Query

Outputs

Configure

Storage account settings

Scale

Locale

Event ordering

Error policy

Compatibility level

Managed Identity

General

Tools

Properties

Monitoring

Metrics

Blob storage

New output

\* Output alias  
blob-telemetry-stream

Provide Blob storage settings manually

Select Blob storage from your subscriptions

Subscription  
Essai gratuit

\* Storage account  
telemetryblobstream

\* Storage account key  
\*\*\*\*\*

\* Container  
Create new Use existing

telemetry-stream

Path pattern  
/

Date format  
YYYY/MM/DD

Time format  
HH

\* Event serialization format  
JSON

Encoding  
UTF-8

Format  
Line separated

Save

The screenshot shows the Azure Stream Analytics job configuration interface. The left sidebar lists various configuration sections: Tags, Diagnose and solve problems, Settings (Locks), Job topology, Inputs, Functions, Query, and Outputs (which is currently selected). The main area displays the 'blob-telemetry-stream' output configuration for blob storage. It includes fields for Output alias (blob-telemetry-stream), Subscription (Essai gratuit), Storage account (telemetryblobstream), Storage account key (redacted), Container (telemetry-stream), Path pattern (/), Date format (YYYY/MM/DD), Time format (HH), Event serialization format (JSON), Encoding (UTF-8), and Format (Line separated). A 'Save' button is located at the bottom right of the configuration pane.

# Stream Analytics job configuration step 9

output : Cosmos DB

The screenshot shows the 'Outputs' section of an Azure Stream Analytics job named 'blob-telemetry-stream'. The sink is configured as 'Blob storage'. On the right, the 'Cosmos DB' configuration pane is open, showing the following fields:

- \* Output alias: cosmos-telemetry-stream
- Provide Cosmos DB settings manually (radio button)
- Select Cosmos DB from your subscriptions (radio button, selected)
- Subscription: Essai gratuit
- \* Account id: bg-cosmos-stream-analytics
- Account key: (redacted)
- \* Database: Create new (radio button, selected)
- Use existing (radio button)
- bg-stream-analytic (selected)
- \* Collection name pattern: telemetry
- Document id: id

A 'Save' button is located at the bottom of the configuration pane.

- Document id: The name of the field in output events used to specify the document id which insert or update operations are based on.  
In our example, we are using the field "id" for that.

# Stream Analytics job configuration step 9

output : Power BI

### netry-stream-analytics - Outputs

| NAME                    | SINK         |
|-------------------------|--------------|
| blob-telemetry-stream   | Blob storage |
| cosmos-telemetry-stream | Cosmos DB    |

**Power BI**  
New output

\* Output alias: powerbi-telemetry-stream ✓

Group workspace: Authorize connection to load workspaces ✓

\* Dataset name: iot-telemetry-stream ✓

\* Table name: telemetry ✓

**Authorize connection**  
You'll need to authorize with Power BI to configure your output settings.

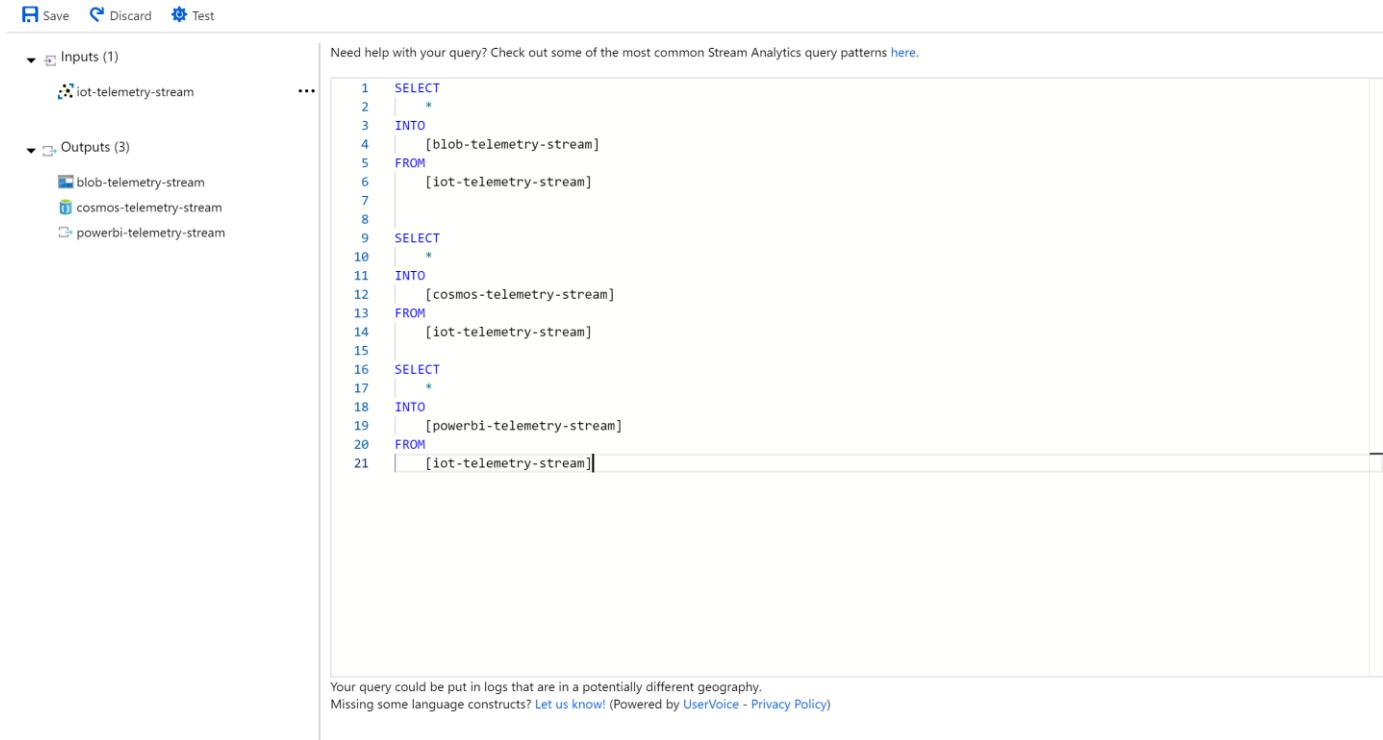
**Authorize**

Don't have a Microsoft Power BI account yet?  
[Sign up](#)

**Note:** You are granting this output permanent access to your Power BI dashboard. Should you need to revoke this access in the future you can do one of the following:  
 1. Change the user account password.  
2. Delete this output.  
3. Delete this job.

**Save**

# Stream Analytics job configuration: query step 10



The screenshot shows the Azure Stream Analytics query editor interface. On the left, there's a sidebar with 'Save', 'Discard', and 'Test' buttons. Below them, the 'Inputs' section shows one item: 'iot-telemetry-stream'. The 'Outputs' section shows three items: 'blob-telemetry-stream', 'cosmos-telemetry-stream', and 'powerbi-telemetry-stream'. The main area contains a query editor with the following T-SQL code:

```
1 SELECT
2   *
3 INTO
4   [blob-telemetry-stream]
5 FROM
6   [iot-telemetry-stream]
7
8
9 SELECT
10  *
11 INTO
12  [cosmos-telemetry-stream]
13 FROM
14  [iot-telemetry-stream]
15
16 SELECT
17  *
18 INTO
19  [powerbi-telemetry-stream]
20 FROM
21  [iot-telemetry-stream]
```

Below the code, a note says: 'Need help with your query? Check out some of the most common Stream Analytics query patterns [here](#).'. At the bottom, a footer message reads: 'Your query could be put in logs that are in a potentially different geography. Missing some language constructs? [Let us know!](#) (Powered by UserVoice - Privacy Policy)'.

# Start Stream Analytics job

step 11

The screenshot shows the Azure Stream Analytics job overview page for the job "bgc-streamAnalytics-telemetry-stream-analytics". The "Start" button is highlighted with a red box. The page displays various details about the job, including its resource group, status, location, and subscription information. It also shows the inputs and outputs defined for the job, along with the query script.

**Resource group:** big-data-classes-telemetry-stream-analytics

**Status:** Stopped

**Location:** West Europe

**Subscription:** Essai gratuit

**Subscription ID:** 7ba392bc-90ba-4f4d-a5ab-3c9e0cc8263c

**Inputs:**

- 1 iot-telemetry-stream

**Outputs:**

- 2 blob-telemetry-stream
- cosmos-telemetry-stream

**Query:**

```
1 SELECT
2 *
3 INTO
4 [blob-telemetry-stream]
5 FROM
6 [iot-telemetry-stream]
7
8
9 SELECT
10 *
11 INTO
12 [cosmos-telemetry-stream]
13 FROM
14 [iot-telemetry-stream]
```

Launch app

step 12

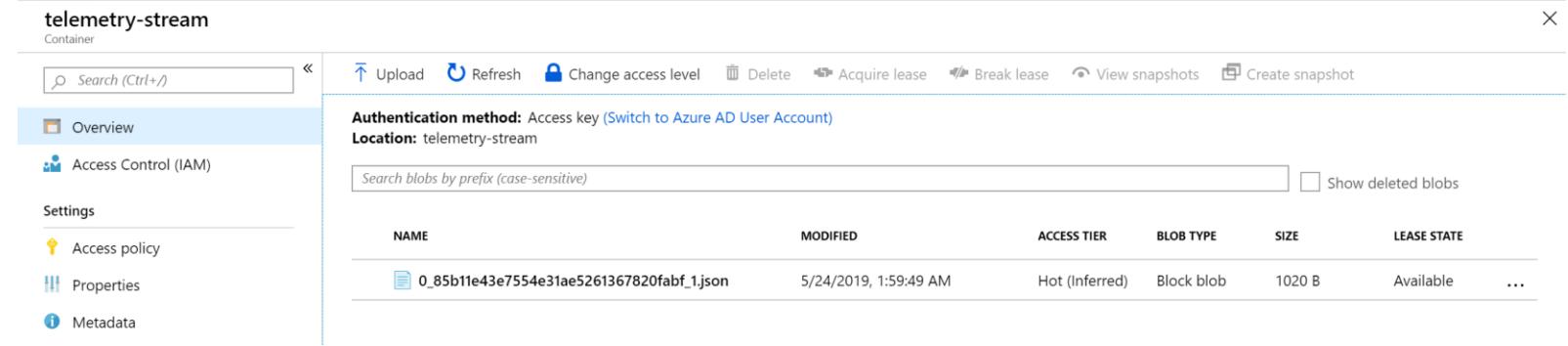
Launch big\_data\_classes\_iot\_stream.js:

```
C:\Users\alexa\Documents\Cours\Big Data\iot>node big_data_classes_iot_stream.js
Client connected
Sending message: {"id":"spaceneedle","temperature":22.80552350706524,"localisation":{"type":"Point","coordinates":[2.348,48.8534]}, "Owner":{"CompanyName":"Miscrosoft", "id_companyName": "4309718283412"}}
send status: MessageEnqueued
Sending message: {"id":"spaceneedle","temperature":24.085870065029816,"localisation":{"type":"Point","coordinates":[2.348,48.8534]}, "Owner":{"CompanyName":"Miscrosoft", "id_companyName": "4309718283412"}}
send status: MessageEnqueued
Sending message: {"id":"spaceneedle","temperature":29.415235594825035,"localisation":{"type":"Point","coordinates":[2.348,48.8534]}, "Owner":{"CompanyName":"Miscrosoft", "id_companyName": "4309718283412"}}
send status: MessageEnqueued
```

# Verify

step 13

Check if the data is sending in blob storage account:



The screenshot shows the Azure Storage Explorer interface. On the left, there's a sidebar with a search bar and navigation links: Overview (selected), Access Control (IAM), Settings (with sub-links: Access policy, Properties, Metadata), and a blob icon. The main area is titled "telemetry-stream Container". It displays a toolbar with Upload, Refresh, Change access level, Delete, Acquire lease, Break lease, View snapshots, and Create snapshot. Below the toolbar, it says "Authentication method: Access key (Switch to Azure AD User Account)" and "Location: telemetry-stream". There's a search bar for blobs by prefix. A table lists a single blob entry:

| NAME                                      | MODIFIED              | ACCESS TIER    | BLOB TYPE  | SIZE   | LEASE STATE |
|---|-----------------------|----------------|------------|--------|-------------|
| 0_85b11e43e7554e31ae5261367820fabf_1.json | 5/24/2019, 1:59:49 AM | Hot (Inferred) | Block blob | 1020 B | Available   |

Blob should be appear in your container.

# Power BI report: creation

step 14

The screenshot shows the Power BI workspace interface. On the left, a sidebar lists navigation options: Favorites, Recent, Apps, Shared with me, Workspaces, My Workspace (which is selected), Dashboards, Reports, Workbooks, and Datasets. The main area displays a search bar and tabs for Dashboards, Reports, Workbooks, and Datasets. Below these tabs is a table header with columns for NAME, ACTIONS, OWNER, and CLASSIFICATION. A message at the bottom states "You don't have any dashboards" and "All dashboards in this workspace will be here." In the top right corner, a "Create" button is visible, with a dropdown menu showing options: Dashboard, Report, Dataset, and Streaming dataset. The "Dataset" option is currently highlighted.

## Power BI report: creation

step 14

### Create report

X

Select a dataset for this report:

 Search

NAME

LAST REFRESHED

iot-telemetry-stream

12 minutes ago

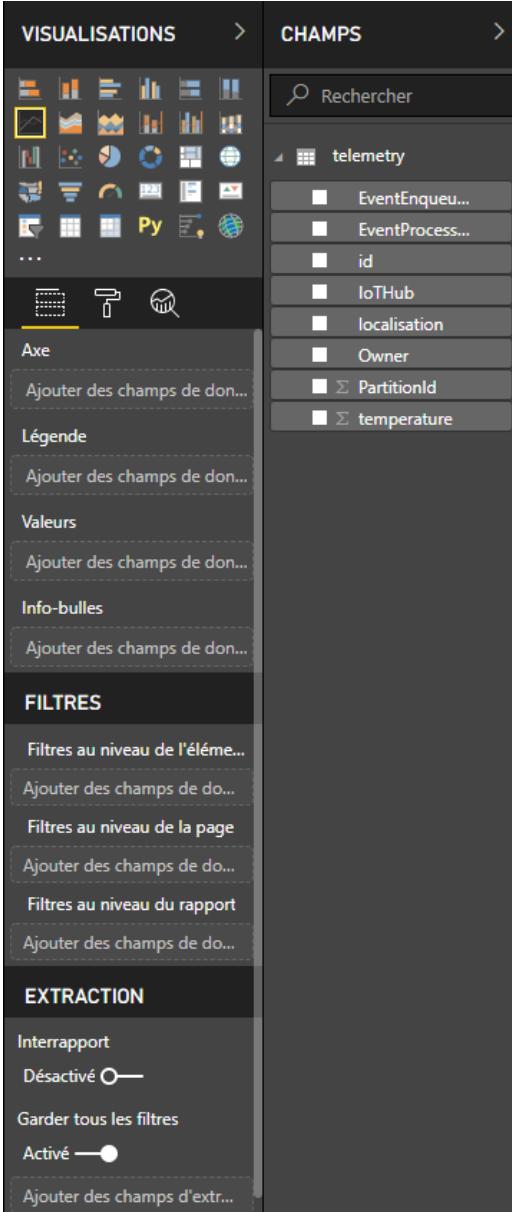
Import new data

Create

Close

# Power BI report: creation

step 14



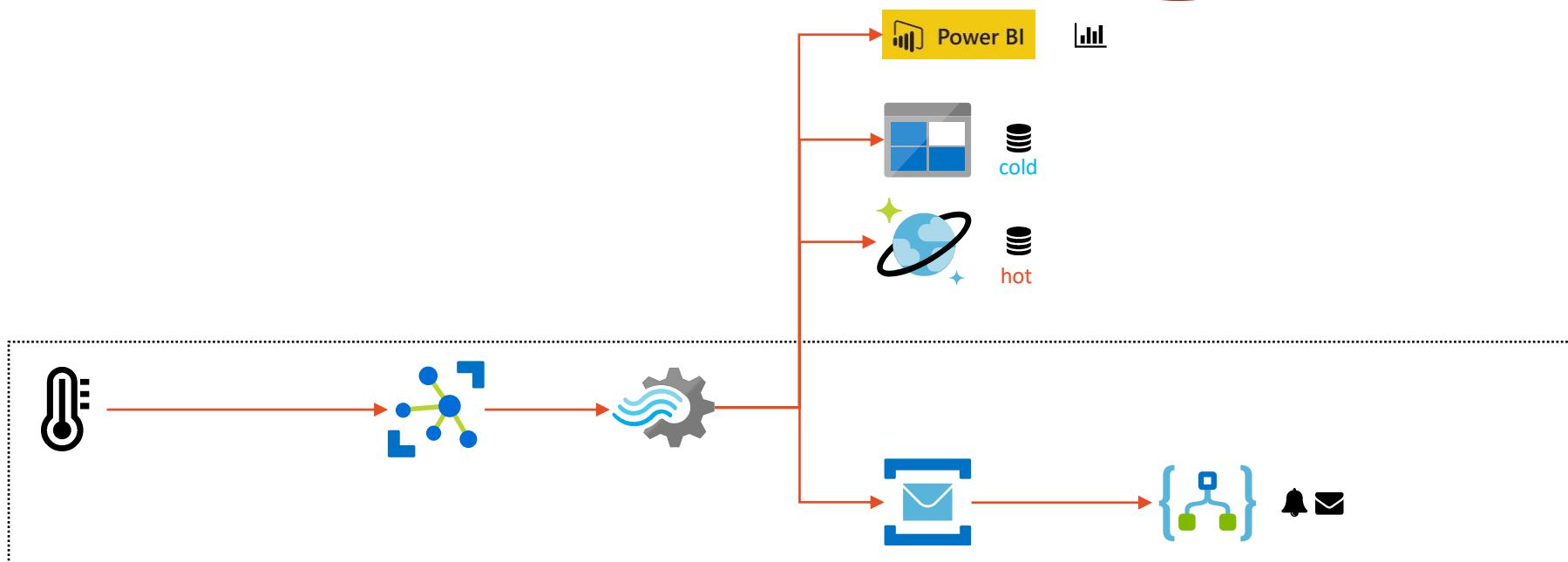
## Add devices

step 15

Add 2 or 3 devices:

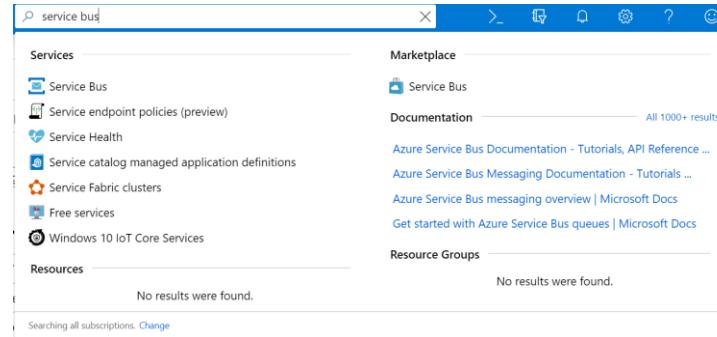
- Copy – paste the “big\_data\_classes\_iot\_stream.js” in multiple files (one for each future device)
- Change the coordinates point and device name.
- Add devices in iot hub
- Use the new connection string in your js file created earlier
- You could see your different device in Power BI

# mail alert

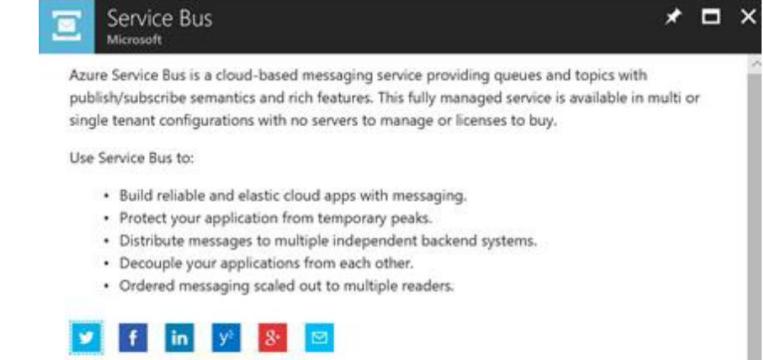


# Service bus creation

## step 1



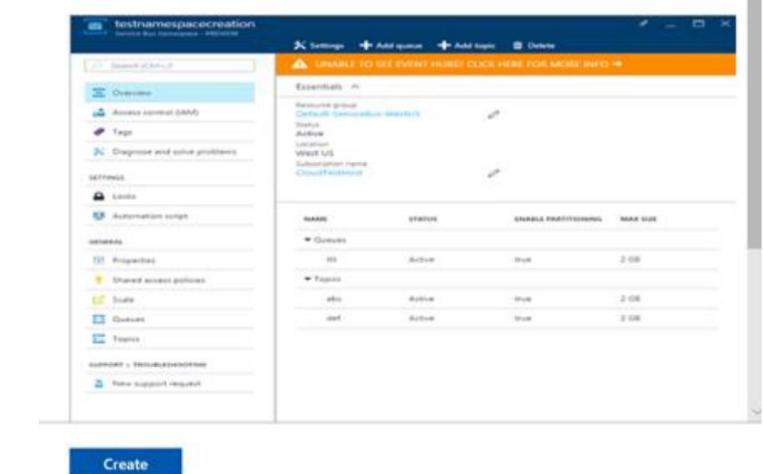
The screenshot shows the Azure portal search results for "service bus". The left sidebar has sections for "Services" (Service Bus, Service endpoint policies (preview), Service Health, Service catalog managed application definitions, Service Fabric clusters, Free services, Windows 10 IoT Core Services) and "Resources" (No results were found). The right sidebar has sections for "Marketplace" (Service Bus), "Documentation" (All 1000+ results, Azure Service Bus Documentation - Tutorials, API Reference ..., Azure Service Bus Messaging Documentation - Tutorials ...), and "Resource Groups" (No results were found). A search bar at the bottom says "Searching all subscriptions. Change".



The screenshot shows the Azure Service Bus overview page. It includes a brief description: "Azure Service Bus is a cloud-based messaging service providing queues and topics with publish/subscribe semantics and rich features. This fully managed service is available in multi or single tenant configurations with no servers to manage or licenses to buy." Below this is a list titled "Use Service Bus to:" with the following items:

- Build reliable and elastic cloud apps with messaging.
- Protect your application from temporary peaks.
- Distribute messages to multiple independent backend systems.
- Decouple your applications from each other.
- Ordered messaging scaled out to multiple readers.

Below the list are social sharing icons for Twitter, Facebook, LinkedIn, YouTube, Google+, and Email.



The screenshot shows the Azure Service Bus management blade for a resource named "testnamespacecreation". The left sidebar has sections for "Essentials" (Resource group, Status, Location, WEST US, Subscription name CloudFrontend), "SETTINGS" (Log, Automation script), "DETAILS" (Properties, Shared access policies, Queue, Queue, Topics), and "SUPPORT + TROUBLESHOOTING" (New support request). The main area displays a table of resources:

| NAME | STATUS | ENABLE PARTITIONING | MAX SIZE |
|------|--------|---------------------|----------|
| test | Active | True                | 2 GB     |
| abc  | Active | True                | 2 GB     |
| def  | Active | True                | 2 GB     |

A blue "Create" button is located at the bottom right of the blade.

# Service bus creation

## step 1

Create namespace □ X

Service Bus

\* Name  
bgc-servicebu-telemetry-stream-analytics ✓  
.servicebus.windows.net

\* Pricing tier ([View full pricing details](#))  
Standard

\* Subscription  
Essai gratuit

\* Resource group  
big-data-classes-telemetry-stream-analytics ✓  
[Create new](#)

\* Location  
West US

**Create**

- Name: choose a different name
- Pricing tier: Standard
- Subscription : choose between available subscriptions
- Resource group: select the previously created group

# Service bus configuration

## step 2

Create a queue:

The screenshot shows the Azure Service Bus Management Portal. On the left, there's a sidebar with navigation links like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Shared access policies, Scale, Geo-Recovery, Migrate to premium), Properties (Locks, Export template), Entities (Queues, Topics), Monitoring (Alerts, Metrics, Diagnostic settings), Support + troubleshooting, and Resource health. The main area shows a resource group named "big-data-classes-telemetry-stream-analytics". A "Create queue" dialog box is open on the right, prompting for a queue name (q1), max queue size (1 GB), message time to live (14 days, 0 hours, 0 minutes, 0 seconds), lock duration (0 days, 0 hours, 0 minutes, 30 seconds), and various optional settings (Enable duplicate detection, Enable dead lettering on message expiration, Enable sessions, Enable partitioning). The "Enable partitioning" checkbox is checked. At the bottom of the dialog is a "Create" button.

# Stream Analytics configuration: output

## step 3

The screenshot shows the Azure Stream Analytics job configuration interface. On the left, the main pane displays the job topology with inputs from 'iot-telemetry-stream' and outputs to 'blob-telemetry-stream'. The right pane is a detailed configuration dialog for the 'blob-telemetry-stream' output.

**Service Bus queue** configuration details:

- Output alias:** servicebus-telemetry-streaming
- Select queue from your subscriptions:** Selected (radio button)
- Subscription:** Essai gratuit
- Service Bus namespace:** bgc-servicebus-telemetry-stream-analytics
- Queue name:** q1 (Use existing selected)
- Queue policy name:** RootManageSharedAccessKey
- Queue policy key:** (redacted)
- Property columns:** (empty)
- Event serialization format:** JSON
- Encoding:** UTF-8

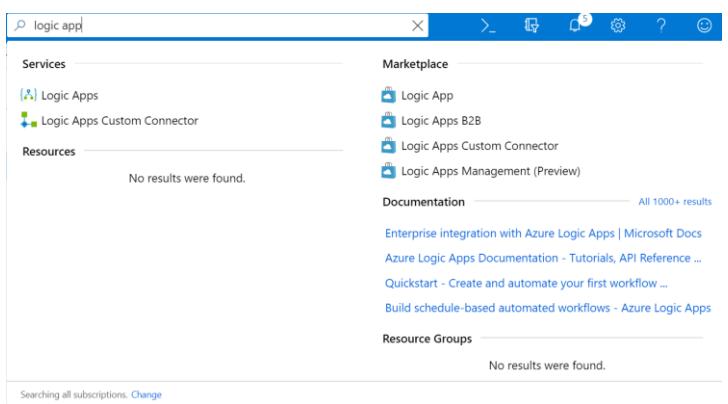
**Query Editor:**

```
1 SELECT
2 *
3 INTO [blob-telemetry-stream]
4 FROM [iot-telemetry-stream]
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

**Note:** Your query could be put in logs that are in a potentially different geography. Missing some language constructs? Let us know! (Powered by UserVoice - Privacy Policy)

# Logic app creation

step 4

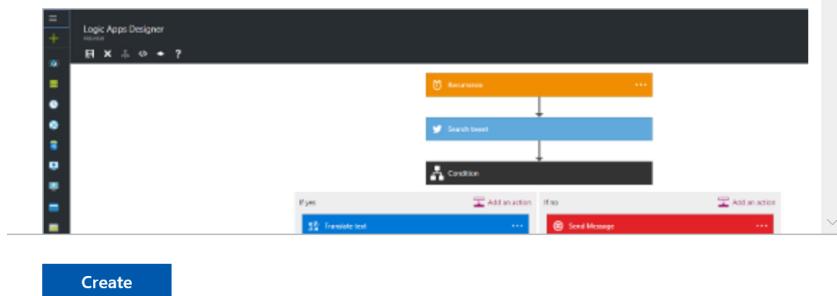


**Easy to use design tools** - Logic Apps can be designed end-to-end in the browser. Start with a trigger - from a simple schedule to whenever a tweet appears about your company. Then orchestrate any number of actions using the rich gallery of connectors.

**Compose SaaS easily** - Even composition tasks that are easy to describe are difficult to implement in code. Logic Apps make it a cinch to connect disparate systems. Want to create a task in CRM based on activity on your Facebook or Twitter accounts? Want to connect your cloud marketing solution to your on-premises billing system? Logic apps are the fastest, most reliable way to deliver solutions to these problems.

**Extensibility baked in** - Don't see the connector you need? Logic Apps are part of the App Service suite and designed to work with API apps; you can easily create your own API app to use as a connector. Build a new app just for you, or share and monetize in the marketplace.

**Real integration horsepower** - Start easy and grow as you need. Logic Apps can easily leverage the power of BizTalk, Microsoft's industry leading integration solution to enable integration professionals to build the solutions they need.



# Logic app creation

step 4

Logic App Create

\* Name  
bgc-logicapp-telemetry-stream-analytics ✓

\* Subscription  
Essai gratuit

\* Resource group i  
 Create new  Use existing  
big-data-classes-telemetry-stream-analytic: ✓

\* Location  
West Europe

Log Analytics i  
On Off

i You can add triggers and actions to your Logic App after creation.

Create Automation options

- Name: choose a different name
- Subscription : choose between available subscriptions
- Resource group: select the previously created group

# Logic app configuration

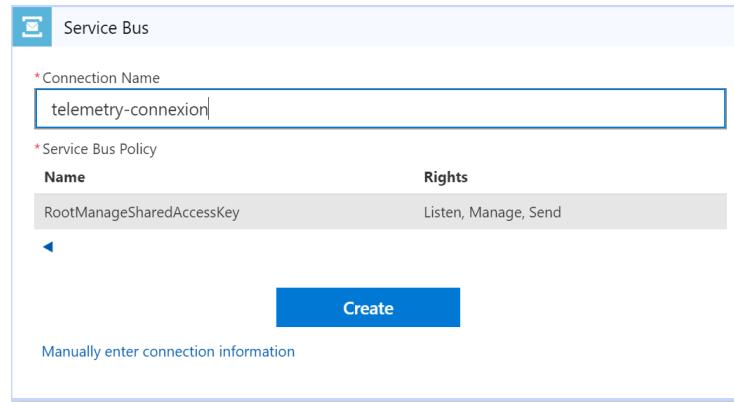
step 5

The screenshot shows the Azure Logic App Designer interface for a logic app named "bgc-logicapp-telemetry-stream-analytics". The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Development Tools (with Logic app designer selected), Logic app code view, Versions, API connections, Quick start guides, Release notes, Settings (Workflow settings, Access keys, Identity, Properties, Locks, Export template), and Monitoring (Alerts). The main area features a video player with the title "Introducing Azure Logic Apps" and a play button. To the right of the video, text reads: "Building integration solutions is easier than ever. Logic Apps brings speed and scalability into the enterprise integration space. The ease of use of the designer, variety of available triggers and actions, and powerful management tools make centralizing your APIs simpler than ever. As businesses move towards digitalization, Logic Apps allows you to connect legacy and cutting-edge systems together." Below the video, a section titled "Start with a common trigger" says "Pick from one of the most commonly used triggers, then orchestrate any number of actions using the rich collection of connectors". A grid of trigger cards is shown, with the first card, "When a message is received in a Service Bus queue", highlighted with a red border.

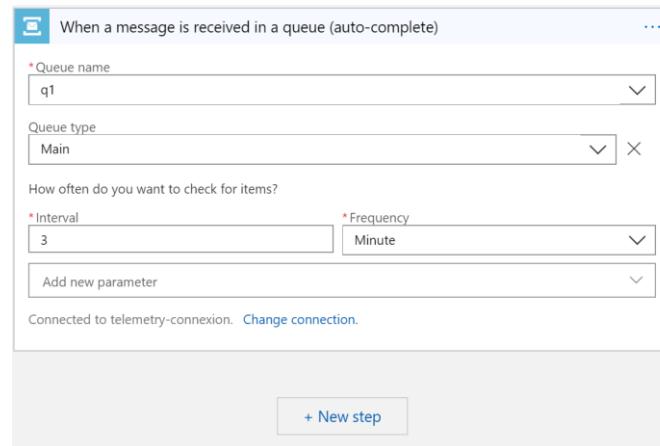
|   |                                    |   |
|---|------------------------------------|---|
| When a message is received in a Service Bus queue | When a HTTP request is received    | When a new tweet is posted                  |
| When an Event Grid resource event occurs          | Recurrence                         | When a new email is received in Outlook.com |
| When a new file is created on OneDrive            | When a file is added to FTP server |   |

# Logic app configuration

step 5



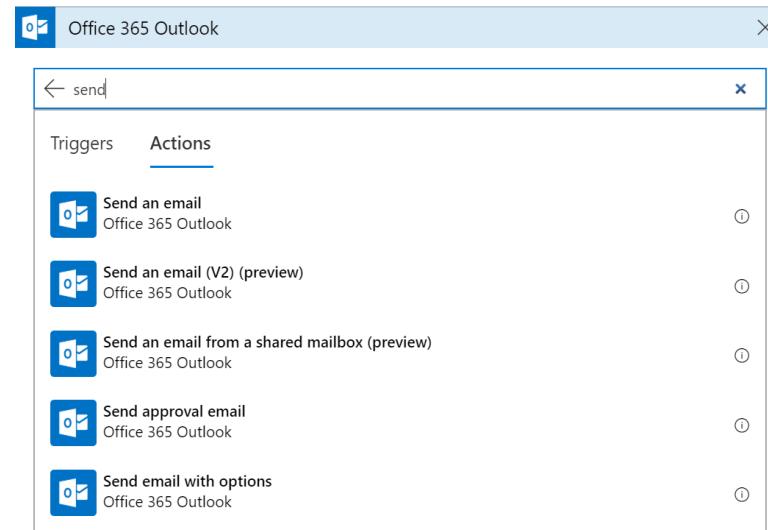
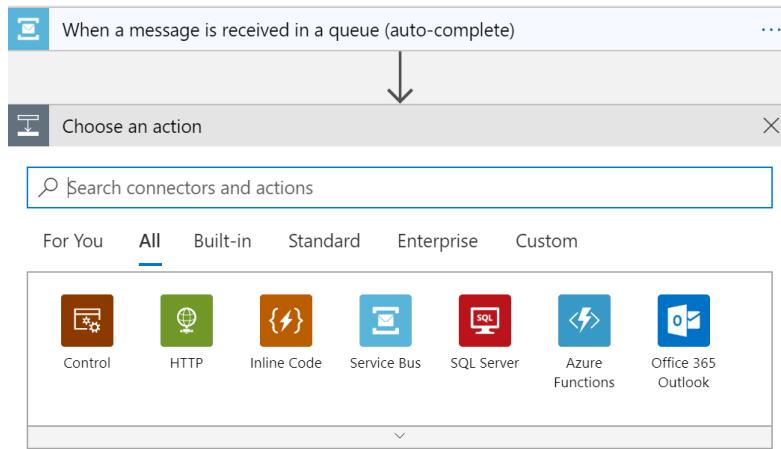
- Click on Create
- Select the service bus you created previously
- Select the RootManageSharedAccessKey policy
- Give a name to the connection



- Select the queue
- Click on Next step and then in Add an action

# Logic app configuration

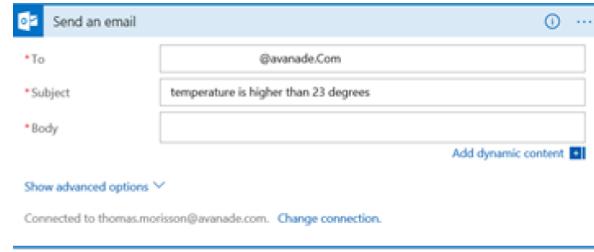
step 5



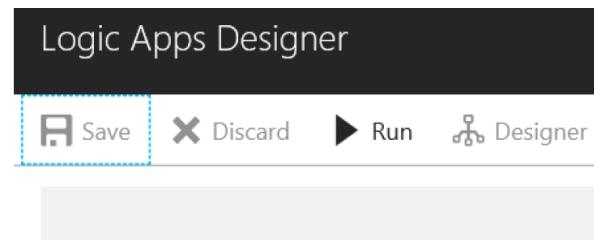
- Click on Next step and then in Add an action
- Look for Office 365 Outlook connector
- Configure the « Send a email » box

# Logic app configuration

step 5



- Configure the « Send a email » box



- Save the workflow, and run it

You can now follow the history of execution

| Runs history   |                         |             |             |
|--|-------------------------|-------------|-------------|
| All  | Start time earlier than | Pick a date | Pick a time |
| Specify the run identifier to open monitor view directly |                         |             |             |
| STATUS   | START TIME              | IDENTIFIER  | DURATION    |
| No runs  |                         |             |             |

| Trigger History                                       |                         |             |
|---|-------------------------|-------------|
| All   | Start time earlier than | Pick a date |
| When_a_message_is_received_in_a_queue_(auto-complete) |                         |             |
| STATUS  | START TIME              | FIRED       |
| Skipped   | 08/02/201...            |             |
| Skipped   | 08/02/201...            |             |