

# Data Engineering in the Cloud

## Real-time Data Processing with Azure Databricks

Xuemao Zhang  
East Stroudsburg University

January 18, 2025

# Outline

- Azure Services Required
- Lab

# Azure Services Required

- Data Sources: Streaming data from IoT devices or social media feeds. We use simulated data in Event Hubs.
- Ingestion: Azure Event Hubs for capturing real-time data.
- Processing: Azure Databricks for stream processing using Structured Streaming.
- Storage: Processed data stored in Azure Data Lake (Delta Format).
- The project is copied from [Real Time Streaming with Azure Databricks and Event Hubs](https://youtu.be/pwWlegHgNRw?si=SP1S9r0Z4AZPzGYA) <https://youtu.be/pwWlegHgNRw?si=SP1S9r0Z4AZPzGYA>

# Azure Services Required

Compared to Azure Stream Analytics, Azure Databricks

- Highly scalable, can handle large volumes of data with Spark's distributed computing power.
- Interactive development environment with notebooks for collaborative data analysis and visualization.
  - ▶ Flexibility: Supports complex transformations and machine learning tasks using Python, Scala, SQL, and R.
- Advanced Analytics: Suitable for advanced analytics, machine learning, and data science use cases.

# Lab

**Step 1.** Create a datalake (storage account with hierarchical namespace enabled)  
- To lower the cost, you may choose Redundancy as LRS - Create a container in the storage account

[Home](#) > [Storage accounts](#) >

## Create a storage account ...

Basics   Advanced   Networking   Data protection   Encryption   Tags   Review + create

[View automation template](#)

### Basics

Subscription	AzureSubscription
Resource group	xzhang2
Location	East US
Storage account name	xzhangstorage2
Performance	Standard
Replication	Read-access geo-redundant storage (RA-GRS)

### Advanced

Enable hierarchical namespace	Enabled
Enable SFTP	Disabled
Enable network file system v3	Disabled
Allow cross-tenant replication	Disabled
Access tier	Hot
Enable large file shares	Enabled

### Security

Secure transfer	Enabled
Blob anonymous access	Disabled
Allow storage account key access	Enabled
Default to Microsoft Entra authorization in the Azure portal	Disabled

[Previous](#)

[Next](#)

[Create](#)

# Lab

**Step 2:** Search for Event Hubs to create an event hub namespace to which the data will be streamed: - Event Hub acts as a message broker for the streaming data.

- 1 Enter the details shown in the image below and click on the “Review + create” button.

Home > Event Hubs >

## Create Namespace

Event Hubs

Basics Advanced Networking Tags Review + create

### Project Details

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

Subscription \* AzureSubscription

Resource group \* xzhang2

[Create new](#)

### Instance Details

Enter required settings for this namespace, including a price tier and configuring the number of units (capacity).

Namespace name \* xzhang2Hubs ✓

.servicebus.windows.net

Location \* East US

**i** The region selected supports Availability zones. Your namespace will have Availability Zones enabled. [Learn more.](#)

Pricing tier \* Basic (~\$11 USD per TU per Month)

[Browse the available plans and their features](#)

Throughput Units \* 1

[Review + create](#) < Previous Next: Advanced >

- 2 Validate the configuration and click on the “Create” button.

[Home](#) > [Event Hubs](#) >

## Create Namespace

Event Hubs

✓ Validation succeeded.

Basics   Advanced   Networking   Tags   Review + create

Event Hubs Namespace  
by Microsoft

Basics

Namespace name	xzhang2Hubs
Subscription	AzureSubscription
Resource group	xzhang2
Location	East US
Pricing tier	Basic
Throughput Units	1
Availability Zones (Zone Redundancy)	Enabled

Networking

Connectivity method	Public access
---------------------	---------------

Security

Minimum TLS version	1.2
Local Authentication	Enabled

Create   < Previous   Next >

- Once the deployment process is completed, click on the “Go to resource” button.

[Home](#) >



xzhang2Hubs | Overview

Deployment

Search

Delete Cancel Redeploy Download Refresh

Overview

Inputs

Outputs

Template

✓ Your deployment is complete



Deployment name : xzhang2Hubs  
Subscription : [AzureSubscription](#)  
Resource group : xzhang2

Start time : 8/2/2024, 10:56:32 AM  
Correlation ID : bb72dd3e-5c8a-4a70-935d-e9d066b7087f

> Deployment details

∨ Next steps

[Go to resource](#)



## 4 The Event Hubs dashboard will be as shown in the image below.

Home >

**xzhang2Hubs** Event Hubs Namespace

Search  + Event Hub Delete Refresh Give feedback

**Overview**

- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Generate data (preview)
- Events
- Settings
  - Shared access policies
  - Scale
  - Geo-Recovery
  - Encryption
  - Configuration
  - Properties
  - Locks
- Entities
- Monitoring
- Automation

**Essentials**

You can start generating test data with the new Azure Event Hubs Data Generator. Click on this message to try the feature!

Resource group (move)	: xzhang2	Created	: Friday, August 2, 2024 at
Status	: Active	Updated	: Friday, August 2, 2024 at
Location	: East US	Zone Redundancy	: Enabled
Subscription (move)	: AzureSubscription	Pricing tier	: Basic
Subscription ID	: c375ef05-a27a-41f9-ba5e-20eebf29c6	Throughput Units	: 1 unit
Host name	: xzhang2Hubs.servicebus.windows.net	Auto-inflate throughput ...	: Not Supported
		Local Authentication	: Enabled

Tags (edit) : Add tags

NAMESPACE CONTENTS 0 EVENT HUBS KAFKA SURFACE NOT SUPPORTED ZONE REDUNDANCY ENABLED

Show data for the last: 1 hour 6 hours 12 hours 1 day 7 days 30 days

Requests	Messages	Throughput
100	100	100B
90	90	90B
80	80	80B
70	70	70B
60	60	60B
50	50	50B

# Lab

- 5 click on the “+ Event Hub” button to create an event hub
- 6 Add a name and select retention period as shown in the image below and click on the “Create” button. (Note down the event hub name for future reference)

[Home](#) > [xzhang2Hubs](#) >

## Create Event Hub

Event Hubs

**Basics** Capture Review + create

### Event Hub Details

Enter required settings for this event hub, including partition count and message retention.

Name \* ⓘ  ✓

Partition count ⓘ  1

### Retention

Configure retention settings for this Event Hub. [Learn more](#)

Cleanup policy ⓘ  ▼

Retention time (hrs) \* ⓘ  min. 1 hour, max. 24 hours (1day)

[Review + create](#) [< Previous](#) [Next: Capture >](#)

## 6 Click Create

Home > xzhang2Hubs >

### Create Event Hub ...

Event Hubs

✓ Validation succeeded.

Basics Capture Review + create

Event Hubs Instance  
by Microsoft

Basics

Name	esuhub1
Partition count	1
Retention	
Cleanup policy	Delete
Retention time (hrs)	1

Capture

Capture Status	Not Supported
----------------	---------------

Create < Previous Next >

- Go to the Settings of the Event Hubs Namespace and go to Shared access policies.
- Update the policy if necessary

The screenshot displays the Azure portal interface for the 'xzhang2Hubs' Event Hubs Namespace. The left-hand navigation pane shows the 'Settings' section expanded, with 'Shared access policies' selected. The main content area shows a table of policies, with 'RootManageSharedAccessKey' highlighted. A side panel on the right provides details for this policy, including the primary and secondary keys, connection strings, and the SAS Policy ARM ID.

**SAS Policy: RootManageSharedAccessKey**

- Save Discard Delete Regenerate Primary Key
- ☒ Manage
- ☒ Send
- ☒ Listen
- Primary key: jn8Pua+J2mGH4uMmJyfyH01U8IZXfmZo+AEhtTW7ws=
- Secondary key: gz0mcmfFW0tgr7OLDuzMdS8jhfZhe03p+AEhOxhj/w=
- Connection string-primary key: Endpoint=sb://xzhang2Hubs.servicebus.windows.net/SharedAccessKeyName=RootMa...
- Connection string-secondary key: Endpoint=sb://xzhang2Hubs.servicebus.windows.net/SharedAccessKeyName=RootMa...
- SAS Policy ARM ID: /subscriptions/c375ef05-a27a-41f9-ba5e-20eeefeb29c6/resourceGroups/xzhang2/prov...

## Step 2: Create an Azure Databricks workspace

[Home](#) > [Azure Databricks](#) >

### Create an Azure Databricks workspace ...

[Basics](#) [Networking](#) [Encryption](#) [Security & compliance](#) [Tags](#) [Review + create](#)

#### Project Details

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

Subscription \* ⓘ

AzureSubscription ▼

Resource group \* ⓘ

xzhang2 ▼

[Create new](#)

#### Instance Details

Workspace name \*

xzhang2db ✓

Region \*

East US 2 ▼

Pricing Tier \* ⓘ

Standard (Apache Spark, Secure with Microsoft Entra ID) ▼

Managed Resource Group name

Enter name for managed resource group

[Review + create](#) [< Previous](#) [Next: Networking >](#)

## 8 Click Create

[Home](#) > [Azure Databricks](#) >

### Create an Azure Databricks workspace ...

✓ Validation Succeeded

#### Basics

Workspace name	xzhang2db
Subscription	AzureSubscription
Resource group	xzhang2
Region	East US 2
Pricing Tier	standard
Managed Resource Group name	

#### Networking

Deploy Azure Databricks workspace with Secure Cluster Connectivity (No Public IP)	No
Deploy Azure Databricks workspace in your own Virtual Network (VNet)	No




#### Encryption

Enable Infrastructure Encryption	No
Enable CMK for Managed Disks	No
Enable CMK for Managed Services	No




#### Security & compliance


- Go to Azure Databricks resource and click on “Launch Workspace”.


Home > xzhang2\_xzhang2db | Overview >


 **xzhang2db**   ...


Azure Databricks Service


Search   Delete 

 Overview

 Activity log

 Access control (IAM)

 Tags

 Diagnose and solve problems

> Settings

> Automation

> Help

^ Essentials

Status : Active

Resource group : [xzhang2](#)

Location : East US 2

Subscription : [AzureSubscription](#)


Subscription ID : c375ef05-a27a-41f9-ba5e-20eebfeb29c6

Tags ([edit](#)) : [Add tags](#)

Managed Resource Group : [data](#)

URL : [https](#)

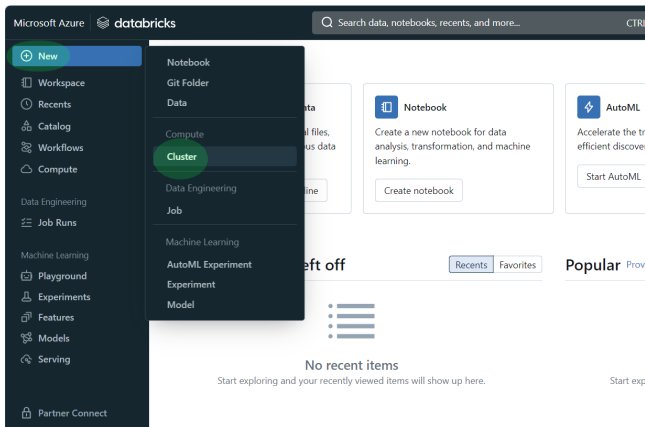
Pricing Tier : [Standard](#)



[Launch Workspace](#)

[Upgrade to Premium](#)

- 10 In the workspace, click on “New Cluster” to create a cluster.





# Lab

1 Settings of the cluster are shown below.

The screenshot shows the Databricks interface for configuring a new compute cluster. The left sidebar contains navigation links: New, Workspace, Recents, Catalog, Workflows, Compute (highlighted), Data Engineering, Job Runs, Machine Learning, Playground, Experiments, Features, Models, and Serving. The main content area is titled 'Compute > New compute >' and shows the configuration for 'DBCluster1'. The 'Multi node' radio button is selected. The 'Access mode' is set to 'Single user access'. The 'Performance' section shows 'Databricks runtime version' as 'Runtime: 14.3 LTS (Scala 2.12, Spark 3.5.0)' and 'Use Photon Acceleration' is checked. The 'Node type' is 'Standard\_DS3\_v2' with '14 GB Memory, 4 Cores'. The 'Terminate after' checkbox is checked, and the value '10' is entered in the minutes of inactivity field. The 'Tags' section has an 'Add tags' button and a table with 'Key' and 'Value' columns. An 'Advanced options' link is at the bottom. At the very bottom, there are 'Create compute' and 'Cancel' buttons.

Microsoft Azure | databricks

Search data, notebooks, recents, and more...

Compute > New compute >

**DBCluster1**

☐ Multi node ☒ Single node

Access mode ⓘ Single user access ⓘ

Single user | v

**Performance**

Databricks runtime version ⓘ

Runtime: 14.3 LTS (Scala 2.12, Spark 3.5.0) | v

☒ Use Photon Acceleration ⓘ

Node type ⓘ

Standard\_DS3\_v2 14 GB Memory, 4 Cores | v ⓘ

☒ Terminate after 10 minutes of inactivity ⓘ

**Tags** ⓘ

Add tags

Key	Value
-----	-------

> Automatically added tags

▶ Advanced options

Create compute Cancel

## 12 Click on "Libraries" and then Click on "Install New"

Microsoft Azure databricks

Search data, notebooks, recents, and more... CTRL + P esudb1

New

- Workspace
- Recents
- Catalog
- Workflows
- Compute
- Data Engineering
- Job Runs
- Machine Learning
- Playground

Compute >

DBCluster1

Configuration Notebooks (0) Libraries Event log Spark UI Driver logs Metrics Apps Spark compute UI - Master

Filter libraries

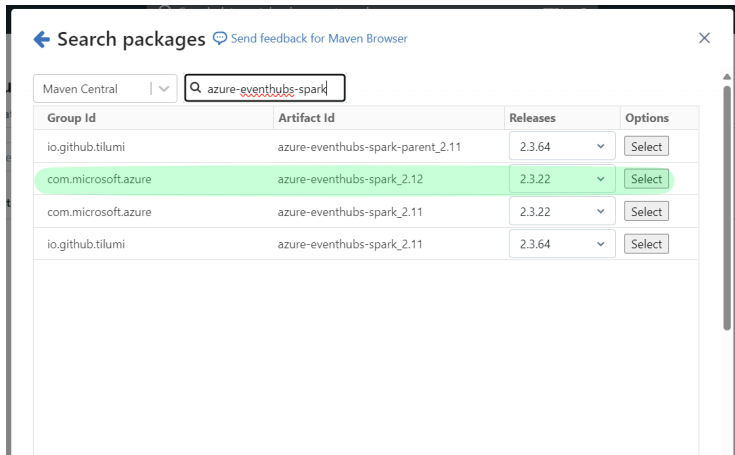
Uninstall Install new

Status	Name	Type	Source
No libraries			

Please install new libraries with Install New

# Lab

- 13 Go to Maven and click on "Search Packages"
- 14 Select Maven Central and search for "azure-eventhubs-spark" and Select "azure-eventhubs-spark\_2.12"



The screenshot shows the Maven Browser search results for the package "azure-eventhubs-spark". The search results are displayed in a table with four columns: Group Id, Artifact Id, Releases, and Options. The second row, corresponding to the package "com.microsoft.azure:azure-eventhubs-spark\_2.12", is highlighted in green. The "Releases" column shows the version "2.3.22" with a dropdown arrow. The "Options" column has a "Select" button. The search bar at the top shows the search term "azure-eventhubs-spark" with a magnifying glass icon.

Group Id	Artifact Id	Releases	Options
io.github.tilumi	azure-eventhubs-spark-parent_2.11	2.3.64	Select
com.microsoft.azure	azure-eventhubs-spark_2.12	2.3.22	Select
com.microsoft.azure	azure-eventhubs-spark_2.11	2.3.22	Select
io.github.tilumi	azure-eventhubs-spark_2.11	2.3.64	Select

# Lab

- 15 After selecting the package, click on "Install"

## Install library [Send feedback for library](#)

Library Source ⓘ

☐ Workspace ☐ File Path/ADLS ☐ PyPI ☒ Maven ☐ CRAN ☐ DBFS

Coordinates

[Search Packages](#)

Repository ⓘ

Exclusions

Cancel

Install

- Go to Azure Databricks workspace, and click on “Compute” and click on the cluster

Microsoft Azure databricks

Search data, notebooks, recents, and more...

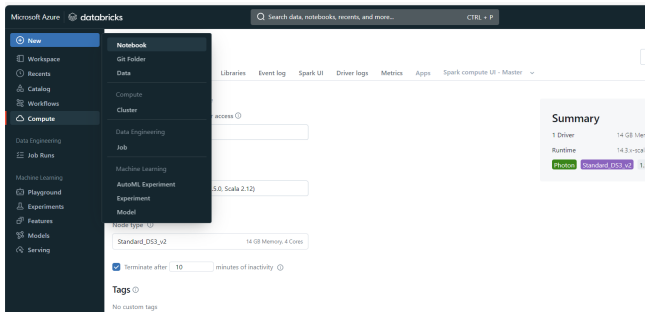
**Compute**

All-purpose compute Job compute Pools

Filter compute you have access to Created by Only pinned

State	Name	Runtime	Active memory	Active co
State icon	DBCluster1	14.3	-	

17 Click on “+ New” -> Notebook to create a new Notebook



# Lab

- 18 Type the code in a cell and run the cell

*#Importing the libraries*

```
from pyspark.sql.functions import *  
from pyspark.sql.types import *
```

Notebook1

Python ▾



File

Edit

View

Run

Help

[Last edit was 2 minutes ago](#)



[Provide feedback](#)



02:20 PM (6s)

1

*#Importing the libraries*

```
from pyspark.sql.functions import *
```

```
from pyspark.sql.types import *
```

## 19 Go to the Event Hubs Namespace and click on the event hub

Home > xzhang2Hubs

**xzhang2Hubs** | Event Hubs ☆ ...  
Event Hubs Namespace

Search

+ Event Hub Refresh Give feedback

Overview  
Activity log  
Access control (IAM)  
Tags  
Diagnose and solve problems  
Generate data (preview)  
Events  
Settings  
Entities  
Event Hubs  
Monitoring  
Automation  
Help

Search to filter items by name...

Name	Status	Message retention
esuhub1	Active	1 hour



## 20 Click Shared access policies

The screenshot shows the Azure portal interface for an Event Hubs instance named 'esuhub1' (xzhang2Hubs/esuhub1). The left-hand navigation pane is expanded, showing the 'Shared access policies' option under the 'Settings' section. The main content area displays the 'Overview' page for this instance. At the top, there is a search bar and a set of action buttons: '+ Consumer group', 'Delete', 'Refresh', and 'Give feedback'. Below this, a message states: 'You can start generating test data with the new Azure Event Hubs Data Generator. Click on this message to...'. The 'Essentials' section lists key properties: Resource group (xzhang2), Location (East US), Subscription (AzureSubscription), Subscription ID (c375ef05-a27a-41f9-ba5e-20eebfeb29c6), and Partition count (1). At the bottom, there are four action cards: 'Capture events' (Use Capture to save your events to persistent storage.), 'Process data' (Process data instantly with Stream Analytics.), 'Checkpoint' (Create consumer groups to checkpoint your events.), and 'Generate data (preview)' (Emit data to your event hub with a few clicks.).

Home >

**esuhub1 (xzhang2Hubs/esuhub1)** ✎ ☆ ...  
Event Hubs Instance

Search ◊ « + Consumer group Delete Refresh Give feedback

**Overview**

- Access control (IAM)
- Diagnose and solve problems
- Settings
  - Shared access policies** ☆
  - Configuration
  - Properties
  - Locks
- Entities
  - Consumer groups
- Features
- Automation
- Help

**Essentials**

Resource group (move) : xzhang2

Location : East US

Subscription (move) : AzureSubscription

Subscription ID : c375ef05-a27a-41f9-ba5e-20eebfeb29c6

Partition count : 1

**Capture events**  
Use Capture to save your events to persistent storage.

**Process data**  
Process data instantly with Stream Analytics.

**Checkpoint**  
Create consumer groups to checkpoint your events.

**Generate data (preview)**  
Emit data to your event hub with a few clicks.

## 21 Create the policy

The screenshot shows the Azure portal interface for an Event Hub instance named 'esuhub1 (xzhang2Hubs/esuhub1)'. The left sidebar contains a navigation menu with options: Overview, Access control (IAM), Diagnose and solve problems, Settings, Shared access policies (selected), Configuration, Properties, Locks, Entities, Consumer groups, Features, Automation, and Help. The main content area displays the 'Shared access policies' section, which is currently empty, showing a message: 'no policies have been set up yet.' On the right, a modal dialog titled 'Add SAS Policy' is open. It has a search bar for 'Policy name' with 'databricks' entered. Below the search bar, there are three checkboxes: 'Manage' (checked), 'Send' (unchecked), and 'Listen' (unchecked). At the bottom of the dialog is a blue 'Create' button.

## 22 Click on the policy and Copy the connection string

The screenshot shows the Azure portal interface for an Event Hubs instance named 'esuhub1 (xzhang2Hubs/esuhub1)'. The left sidebar contains navigation options: Overview, Access control (IAM), Diagnose and solve problems, Settings, Shared access policies (selected), Configuration, Properties, Locks, Entities, Consumer groups, Features, Automation, and Help. The main content area displays the 'Shared access policies' for the instance, with a search bar and a list of policies. The 'databricks' policy is highlighted with a green circle. A side panel titled 'SAS Policy: databricks' is open, showing the following details:

- Primary key:** yWlWfntb6DndAVG/Wrbzrma6WWY13pNV+AEKqf0Xds=
- Secondary key:** MYUHWmPLatSp5WVQ01dnqpeTPQ1IKYG+AEHD01tZE=
- Connection string - primary key:** Endpoint=sb://xzhang2Hubs.servicebus.windows.net/SharedAccessKeyName=databricks=
- Connection string - secondary key:** Endpoint=sb://xzhang2Hubs.servicebus.windows.net/SharedAccessKeyName=databricks=
- SAS Policy ARM ID:** /subscriptions/c375af05-a27a-41f9-ba5e-20aebf6b29cd/resourceGroups/xzhang2/prov...

# Lab

- 23 Go back to the notebook in the databricks

```
# create a schema
spark.sql("create schema streaming;")
```

```
#Set up Azure Event hubs connection string
#Replace with your Event Hub namespace, name, and key
connectionString = "Endpoint=sb://xzhang2hubs.servicebus.windows.net/;
SharedAccessKeyName=databricks;
SharedAccessKey=yWtWfhtb90hdAYGi9Wtrbzmw8WWYI3pNV+AEhKgRKdo=;
EntityPath=esuhub1"
eventHubName = "esuhub1"

ehConf = {
  'eventhubs.connectionString' : sc._jvm.org.apache.spark.eventhubs.EventHubsUtils.e
  'eventhubs.eventHubName': eventHubName
}
```

# Lab

- We can display the contents of a DBFS mount point or directory in Azure Databricks by running

```
# Using shell command to list files
```

```
files = dbutils.fs.ls("/mnt")
```

```
# Display the files in a table format
```

```
display(files)
```

lecture26 Python ☆

File Edit View Run Help [Last edit was now](#) [Provide feedback](#) [Run all](#)

```
# Using shell command to list files
files = dbutils.fs.ls("/mnt")

# Display the files in a table format
display(files)
```

▶ (2) Spark Jobs

Table +

	path	name	size	modificationTime
1	dbfs/mnt/streaming/	streaming/	0	1722625851000

2 minutes ago (2s)

2

```
# create a schema
spark.sql("create schema streaming;")
```

DataFrame[]

1 minute ago (<1s)

3

Python

```
#Set up Azure Event hubs connection string
#Replace with your Event Hub namespace, name, and key
connectionString = "Endpoint=sb://xzhang2hubs.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;
SharedAccessKey=jnBPua+J2mGHJwMmJyFyH01U81ZKjFm2o+AEhLTW7ws="
eventHubName = "esuhub1"

ehConf = {
    'eventhubs.connectionString': sc._jvm.org.apache.spark.eventhubs.EventHubsUtils.encrypt(connectionString),
    'eventhubs.eventHubName': eventHubName
}
```

# Lab

24 Run the following cell

- Send data from the event hub while the stream is running

```
{  
  "temperature": 20,  
  "humidity": 60,  
  "windSpeed": 10,  
  "windDirection": "NW",  
  "precipitation": 0,  
  "conditions": "Partly Cloudy"  
}
```

```
# Reading stream: Load data from Azure Event Hub into DataFrame 'df' using the previous cell  
df = spark.readStream \  
  .format("eventhubs") \  
  .options(**ehConf) \  
  .load() \  
  
# Displaying stream: Show the incoming streaming data for visualization and debugging  
df.display()
```

Notebook1

Python

☆

File

Edit

View

Run

Help

Last edit was now

Provide feedback

Interrupt

DBCluster1

Scd

Interrupt

4

Python

.load() \

# Displaying stream: Show the incoming streaming data for visualization and debugging purposes

df.display()

(1) Spark Jobs

display\_query\_12 (id: 332d88bd-0d55-4fd0-b8e0-942a2d887a61) Last updated: 10 seconds ago

Dashboard

Raw Data

{

"id" : "332d88bd-0d55-4fd0-b8e0-942a2d887a61",

"runId" : "e91f7ata-c698-4372-8372-101dfida99b",

"name" : "display\_query\_12",

"timestamp" : "2024-08-02T19:11:59.000Z",

"batchId" : 2,

"batchDuration" : 10,

"numInputRows" : 0,

"inputRowsPerSecond" : 0.0,

"processedRowsPerSecond" : 0.0,

"durationMs" : {

"getOffset" : 10,

"endOfExecution" : 10,

}

}

Table

+

110

body

1

> evogjCagInRbBcmf0dXijogMjAsCAgCAaHvWpdpHcOA2MCwKCAgCj3aW5ktJ8Z2WQOiaAMCwKCAgCj3aW5kGyZWN0...

partition0

offset816

sequenceNumber



# Lab

## 25 Writing stream

*# Writing stream: Persist the streaming data to a Delta table 'streaming.weather'*

```
df.writeStream\  
  .option("checkpointLocation", "/mnt/streaming/weather")\  
  .outputMode("append")\  
  .format("delta")\  
  .toTable("streaming.weather")
```

The screenshot shows a Databricks notebook interface. At the top, there's a toolbar with an 'Interrupt' button and a cell number '5'. The code cell contains the following Python code:

```
df.writeStream\  
  .option("checkpointLocation", "/mnt/streaming/weather")\  
  .outputMode("append")\  
  .format("delta")\  
  .toTable("streaming.weather")
```

Below the code cell, a Spark job is listed with the ID '63e31f46-3584-4431-bd07-cb1659a6bca7' and a status of 'Last updated: 5 seconds ago'. The job is expanded, showing a 'Dashboard' and 'Run Data' tab. The 'Run Data' tab displays a JSON object with the following details:

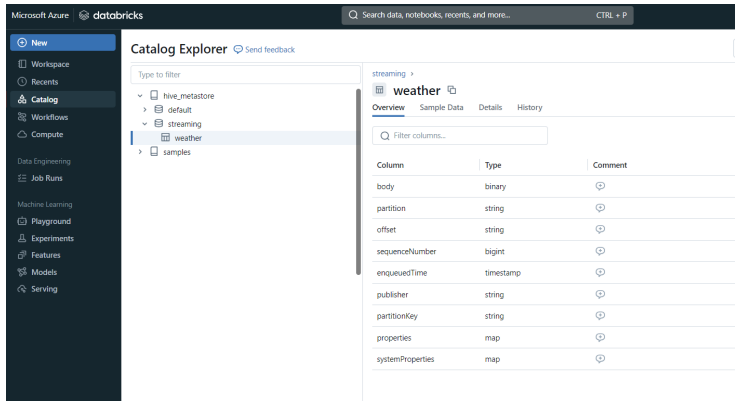
```
{  
  "id": "63e31f46-3584-4431-bd07-cb1659a6bca7",  
  "runId": "b780ca8f-0bdf-4e15-b034-f204a15035c5",  
  "name": null,  
  "timestamp": "2024-08-02T19:13:53.298Z",  
  "batchId": 1,  
  "batchDuration": 9,  
  "numInputRows": 0,  
  "inputRowsPerSecond": 0.0,  
  "processedRowsPerSecond": 0.0,  
  "durationMs": {  
    "getOffset": 9,  
    "endTimeExecution": 0  
  }  
}
```

At the bottom of the job details, the query ID is shown: 'cpyspark.sql.streaming.query.StreamingQuery at 0x7f293c76dbd0'.

In Spark Structured Streaming, the `outputMode` specifies how the results of a streaming query should be written to the output sink. There are three output modes: `append`, `complete`, and `update`. Each mode has different behaviors and use cases:

- **Append Mode:** Only the new rows that are appended to the result table since the last trigger are written to the sink
- **Complete Mode:** The entire result table is written to the sink every time there is a trigger. This means all rows of the result are written out, not just the new rows.
- **Update Mode:** Only the rows that have changed since the last trigger are written to the sink. This includes both newly added rows and updated row

- 26 Check the Catalog, you should be able to see the table `weather`



The screenshot shows the Databricks Catalog Explorer interface. On the left, a sidebar contains navigation options: New, Workspace, Recents, Catalog, Workflows, Compute, Data Engineering, Job Runs, Machine Learning, Playground, Experiments, Features, Models, and Serving. The 'Catalog' option is selected. The main area is titled 'Catalog Explorer' and includes a search bar 'Type to filter'. Below the search bar, a tree view shows the catalog structure: 'hive\_metastore' (expanded) contains 'default' (expanded), which contains 'streaming' (expanded), which contains 'weather' (selected) and 'samples'. The right pane shows the 'weather' table details. It has tabs for 'Overview', 'Sample Data', 'Details', and 'History'. The 'Overview' tab is active, showing a 'Filter columns...' search bar and a table of columns:

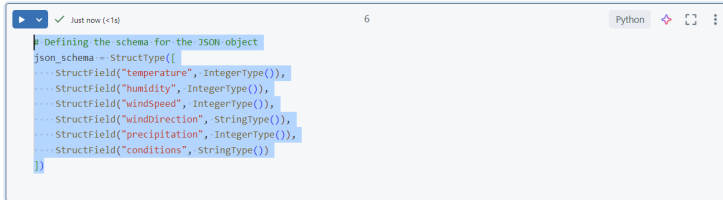
Column	Type	Comment
body	binary	
partition	string	
offset	string	
sequenceNumber	bigint	
enqueuedTime	timestamp	
publisher	string	
partitionKey	string	
properties	map	
systemProperties	map	

# Lab

## 27 Defining the schema for the JSON object.

*# Defining the schema for the JSON object*

```
json_schema = StructType([  
    StructField("temperature", IntegerType()),  
    StructField("humidity", IntegerType()),  
    StructField("windSpeed", IntegerType()),  
    StructField("windDirection", StringType()),  
    StructField("precipitation", IntegerType()),  
    StructField("conditions", StringType())  
])
```



The screenshot shows a Jupyter Notebook interface. At the top, there's a status bar with a play button, a checkmark, the text 'Just now (<1s)', a line number '6', and icons for 'Python', a diamond, a square, and a vertical ellipsis. Below this is a code cell with the following text:   
| Defining the schema for the JSON object  
| json\_schema = StructType([  
| ... StructField("temperature", IntegerType()),  
| ... StructField("humidity", IntegerType()),  
| ... StructField("windSpeed", IntegerType()),  
| ... StructField("windDirection", StringType()),  
| ... StructField("precipitation", IntegerType()),  
| ... StructField("conditions", StringType())  
| ])

## 28 Reading and Transforming

*# Reading and Transforming: Load streaming data from the 'streaming.weather' Delta table, cast 'body' to string*

```
df = spark.readStream\  
  .format("delta")\  
  .table("streaming.weather")\  
  .withColumn("body", col("body").cast("string"))\  
  .withColumn("body", from_json(col("body"), json_schema))\  
  .select("body.temperature", "body.humidity", "body.windSpeed",  
         "body.windDirection", "body.precipitation",  
         "body.conditions", col("enqueuedTime").alias('timestamp'))
```

*# Displaying stream: Visualize the transformed data in the DataFrame for verification and analysis*

```
df.display()
```

*# Writing stream: Save the transformed data to the 'streaming.weather2' Delta table in 'append' mode*

```
df.writeStream\  
  .option("checkpointLocation", "/mnt/streaming/weather2")\  
  .outputMode("append")\  
  .format("delta")\  
  .toTable("streaming.weather2")
```

**Notebook1** Python ☆  
File Edit View Run Help Last edit was 2 minutes ago Provide feedback Interrupt DBCluster1 Schedule

Interrupt 7 Python

```
# Reading and Transforming: Load streaming data from the 'streaming.weather' Delta table, cast 'body' to string, parse JSON, and select specific fields
df = spark.readStream\
    .format("delta")\
    .table("streaming.weather")\
    .withColumn("body", col("body").cast("string"))\
    .withColumn("body", from_json(col("body"), json_schema))\
    .select("body.temperature", "body.humidity", "body.windSpeed", "body.windDirection", "body.precipitation", "body.conditions", col("enqueuedTime").alias('timestamp'))

# Displaying stream: Visualize the transformed data in the DataFrame for verification and analysis
df.display()

# Writing stream: Save the transformed data to the 'streaming.weather2' Delta table in 'append' mode with checkpointing for data reliability
df.writeStream\
    .option("checkpointLocation", "/mnt/streaming/weather2")\
    .outputMode("append")\
    .format("delta")\
    .toTable("streaming.weather2")
```

(2) Spark Jobs

display\_query\_13 (id: b78db3ed-d2a4-4216-b2c3-478ab098d7a4) Last updated: 10 seconds ago

1c0d9e3a-09be-4ff8-9cf9-d0face2a860b Last updated: 5 seconds ago

## 29 Check the Catalog again

Microsoft Azure databricks

Search data, notebooks, recents, and more... CTRL + P

New

- Workspace
- Recents
- Catalog**
- Workflows
- Compute
- Data Engineering
- Job Runs
- Machine Learning
- Playground
- Experiments
- Features
- Models
- Serving
- Partner Connect

Catalog Explorer [Send feedback](#)

Type to filter

- hive\_metastore
  - default
  - streaming
    - weather
    - weather2**
  - samples

streaming >

**weather2**

Overview Sample Data Details History

Filter columns...

Column	Type	Comment
temperature	int	
humidity	int	
windSpeed	int	
windDirection	string	
precipitation	int	
conditions	string	
timestamp	timestamp	

## 30 Reading, aggregating and writing the stream

*# Aggregating Stream: Read from 'streaming.silver.weather', apply watermarking and windowing, and*

```
df = spark.readStream\  
  .format("delta")\  
  .table("streaming.weather2")\  
  .withWatermark("timestamp", "1 minutes") \  
  .groupBy(window("timestamp", "1 minutes")) \  
  .agg(avg("temperature").alias('temperature'), avg("humidity").alias('humidity'),  
       avg("windSpeed").alias('windSpeed'), avg("precipitation").alias('precipitation'))\  
  .select('window.start', 'window.end', 'temperature', 'humidity', 'windSpeed', 'precipitation')
```

*# Displaying Aggregated Stream: Visualize aggregated data for insights into weather trends*

```
df.display()
```

*# Writing Aggregated Stream: Store the aggregated data in 'streaming.gold.weather\_aggregated' with*

```
df.writeStream\  
  .option("checkpointLocation", "/mnt/streaming/weather_summary")\  
  .outputMode("append")\  
  .format("delta")\  
  .toTable("streaming.weather_summary")
```



Notebook1

Python

☆

File

Edit

View

Run

Help

Last edit was now

Provide feedback

Interrupt

DBCluster1

03:26 PM

8

Python

```

df = spark.readStream\
    .format("delta")\
    .table("streaming.weather2")\
    .withWatermark("timestamp", "2 minutes") \
    .groupBy(window("timestamp", "2 minutes")) \
    .agg(avg("temperature").alias('temperature'), avg("humidity").alias('humidity'), avg("windSpeed").alias('windSpeed'), avg("precipitation").alias('precipitation'))\
    .select('window.start', 'window.end', 'temperature', 'humidity', 'windSpeed', 'precipitation')

# Displaying Aggregated Stream: Visualize aggregated data for insights into weather trends
df.display()

# Writing Aggregated Stream: Store the aggregated data in 'streaming.gold.weather_aggregated' with checkpointing for data integrity
df.writeStream\
    .option("checkpointLocation", "/mnt/streaming/weather_summary")\
    .outputMode("append")\
    .format("delta")\
    .toTable("streaming.weather_summary")

```

(2) Spark Jobs

display\_query\_14 (id: 0dd7581b-ba8c-48f7-b58d-bba46c79de50) Last updated: 13 minutes ago

9445d512-acd6-4be0-b14a-3f877690cbb5 Last updated: 12 minutes ago

Table

end

1.2 temperature

1.2 humidity

1.2 windSpeed

1.2 precipitation

	start	end	1.2 temperature	1.2 humidity	1.2 windSpeed	1.2 precipitation
1	2024-08-02T19:25:00.000+00:...	2024-08-02T19:30:00.000+00:...	20	60	10	0
2	2024-08-02T19:30:00.000+00:...	2024-08-02T19:35:00.000+00:...	21	50	20	0
3	2024-08-02T19:20:00.000+00:...	2024-08-02T19:25:00.000+00:...	20	60	10	0

Xuemao Zhang East Stroudsburg University

Data Engineering in the Cloud

January 18, 2025

41 / 47

## 31 Check the catalog again

The screenshot shows the Databricks Catalog Explorer interface. On the left, a sidebar contains navigation options: New, Workspace, Recents, Catalog (selected), Workflows, Compute, Data Engineering, Job Runs, Machine Learning, and Playground. The main area is titled 'Catalog Explorer' and includes a search bar and a 'Send feedback' link. Below the search bar, a tree view shows the catalog structure: hive\_metastore > default > streaming > weather > weather\_summary. The 'weather\_summary' table is selected. To the right, the 'Sample Data' tab is active, displaying a table with 3 rows and 5 columns: start, end, 1.2 temperature, 1.2 humidity, and 1. The data is as follows:

	start	end	1.2 temperature	1.2 humidity	1
1	2024-08-02T19:20:00.000+00:00	2024-08-02T19:25:00.000+00:00	20	60	
2	2024-08-02T19:30:00.000+00:00	2024-08-02T19:35:00.000+00:00	21	50	
3	2024-08-02T19:25:00.000+00:00	2024-08-02T19:30:00.000+00:00	20	60	

## 32 Summary of the three steps

```
json_schema = StructType([
    StructField("temperature", IntegerType()),
    StructField("humidity", IntegerType()),
    StructField("windSpeed", IntegerType()),
    StructField("windDirection", StringType()),
    StructField("precipitation", IntegerType()),
    StructField("conditions", StringType())
])
```

*# Reading stream: Load data from Azure Event Hub into DataFrame 'df' using the previously configured schema*

```
df1 = spark.readStream \
    .format("eventhubs") \
    .options(**ehConf) \
    .load() \
```

*# Writing stream: Persist the streaming data to a Delta table 'streaming.weather' in 'append' mode*

```
df1.writeStream\
    .option("checkpointLocation", "/mnt/streaming/weather")\
    .outputMode("append")\
    .format("delta")\
    .toTable("streaming.weather")
```

# Lab

```
# Reading and Transforming: Load streaming data from the 'streaming.weather' Delta table, cast 'body' to string
df2 = spark.readStream\
    .format("delta")\
    .table("streaming.weather")\
    .withColumn("body", col("body").cast("string"))\
    .withColumn("body", from_json(col("body"), json_schema))\
    .select("body.temperature", "body.humidity", "body.windSpeed", "body.windDirection",
            "body.precipitation", "body.conditions", col("enqueuedTime").alias('timestamp'))

# Displaying stream: Visualize the transformed data in the DataFrame for verification and analysis
df2.display()

# Writing stream: Save the transformed data to the 'streaming.weather2' Delta table in 'append' mode
df2.writeStream\
    .option("checkpointLocation", "/mnt/streaming/weather2")\
    .outputMode("append")\
    .format("delta")\
    .toTable("streaming.weather2")
```

# Lab

*# Aggregating Stream: Read from 'streaming.silver.weather', apply watermarking and windowing, and aggregate*

```
df3 = spark.readStream\  
  .format("delta")\  
  .table("streaming.weather2")\  
  .withWatermark("timestamp", "2 minutes") \  
  .groupBy(window("timestamp", "2 minutes")) \  
  .agg(avg("temperature").alias('temperature'), avg("humidity").alias('humidity'),  
        avg("windSpeed").alias('windSpeed'), avg("precipitation").alias('precipitation'))\  
  .select('window.start', 'window.end', 'temperature', 'humidity', 'windSpeed', 'precipitation')
```

*# Displaying Aggregated Stream: Visualize aggregated data for insights into weather trends*

```
df3.display()
```

*# Writing Aggregated Stream: Store the aggregated data in 'streaming.gold.weather\_aggregated' with delta format*

```
df3.writeStream\  
  .option("checkpointLocation", "/mnt/streaming/weather_summary")\  
  .outputMode("append")\  
  .format("delta")\  
  .toTable("streaming.weather_summary")
```

## lecture26 Python ☆

File Edit View Run Help [Last edit was now](#) [Provide feedback](#)

▶ Run all



▶ ✓ Just now (&lt;1s)

1

Python



```
# Using shell command to list files
files = dbutils.fs.ls("/mnt/streaming")

# Display the files in a table format
display(files)
```

▶ (2) Spark Jobs

Table ▾ +



	path	name	size	modificationTime
1	> dbfs:/mnt/streaming/...	weather/	0	1722625851000
2	> dbfs:/mnt/streaming/...	weather2/	0	1722626475000
3	> dbfs:/mnt/streaming/...	weather_summar...	0	1722626777000

↓ 3 rows | 0.50 seconds runtime

Refreshed now

# License



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](#).