

ASSESSMENT

NAME-SWARNA LATHA GUVVALA

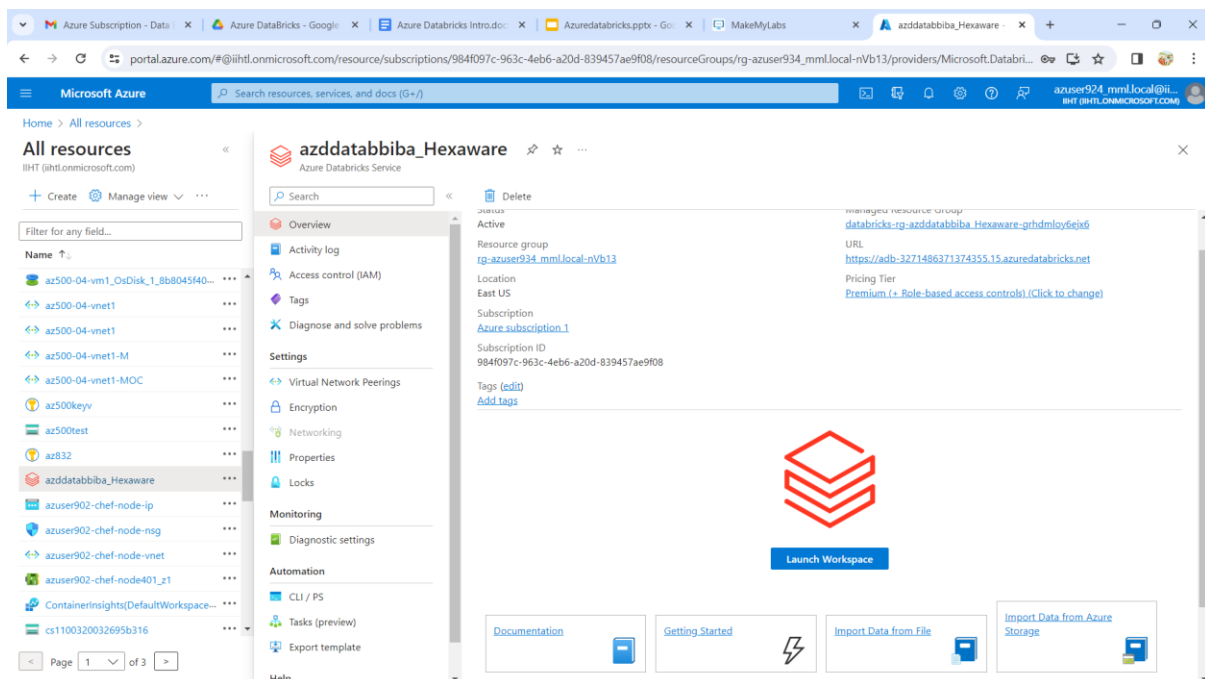
DATE-29/12/2023

CREATE AN AZURE DATABRICKS WORKSPACE:

- In the Azure portal, select **Create a resource > Analytics > Azure Databricks**.
- Under **Azure Databricks Service**, provide the values to create a Databricks workspace.


Property	Description
Workspace name	Provide a name for your Databricks workspace
Subscription	From the drop-down, select your Azure subscription.
Resource group	Specify whether you want to create a new resource group or use an existing one. A resource group is a container that holds related resources for an Azure solution. For more information, see Azure Resource Group overview .
Location	Select West US 2 . For other available regions, see Azure services available by region .
Pricing Tier	Choose between Standard , Premium , or Trial . For more information on these tiers, see Databricks pricing page .

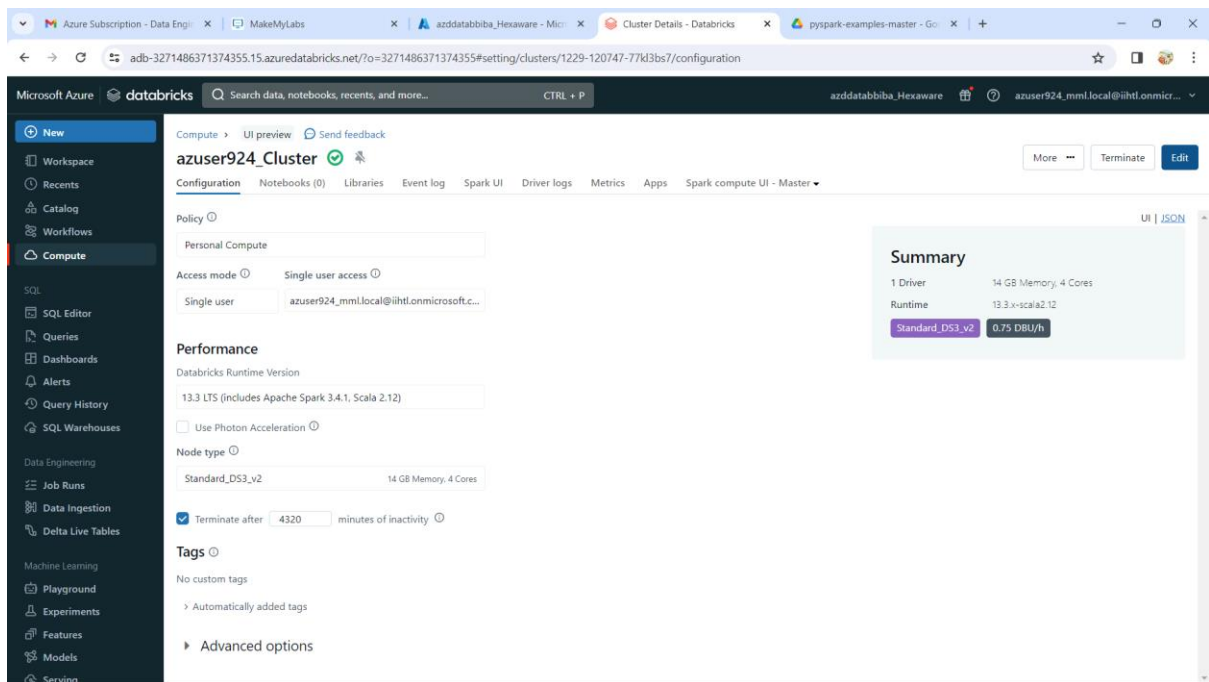
- Select **Review + Create**, and then **Create**. The workspace creation takes a few minutes. During workspace creation, you can view the deployment status in **Notifications**. Once this process is finished, your user account is automatically added as an admin user in the workspace.



CREATE CLUSTER:



A cluster is a collection of Azure Databricks computation resources. To create a cluster:

1. In the sidebar, click  **Compute**.
2. On the Compute page, click **Create Compute**.
3. On the New Compute page, select **13.3 LTS (Scala 2.12, Spark 3.3.2)** or higher from the Databricks Runtime version dropdown.
4. Click **Create Cluster**.



CREATE NOTEBOOK:

A notebook is a collection of cells that run computations on an Apache Spark cluster. For more information on using notebooks, see [Introduction to Databricks notebooks](#). To create a notebook in the workspace:

1. In the sidebar, click  **Workspace**.
2. In your Home  folder, click the blue **Add** button > **Notebook**.
3. Replace the default name of your notebook with your own title and select **SQL** in the language drop-down. This selection determines the *default language* of the notebook.



4. Attach the notebook to the cluster you created. Click the cluster selector in the notebook toolbar and select your cluster from the dropdown menu. If you don't see your cluster, click **More...** and select the cluster from the dropdown menu in the dialog.

PRACTICE PROGRAMS:

The screenshot shows the Databricks notebook interface. The notebook is titled "python_Notebook 2023-12-29 17:42:46". The code in the notebook is as follows:

```
65 spark.sql("select _nullsafeUDF(Name) from NAME_TABLE2") \
66   .show(truncate=False)
67
68 spark.sql("select Seqno, _nullsafeUDF(Name) as Name from NAME_TABLE2 " + \
69   " where Name is not null and _nullsafeUDF(Name) like '%John%'" ) \
70   .show(truncate=False)
```

The output of the code is displayed below the code cell. It shows the results of the SQL queries. The first query returns a single row with the value of the `_nullsafeUDF(Name)` function. The second query returns a table with two columns: `Seqno` and `Name`.

Seqno	Name
1	John Jones

Command took 14.85 seconds -- by azuser924_mml.local@ihti.onmicrosoft.com at 12/29/2023, 5:49:05 PM on azuser924_Cluster

The screenshot shows the Databricks notebook interface. The notebook is titled "python_Notebook 2023-12-29 17:42:46". The code in the notebook is as follows:

```
42
df: pyspark.sql.dataframe.DataFrame = [firstname: string, middlename: string ... 4 more fields]
StringType()
IntegerType()
True
root
|-- firstname: string (nullable = true)
|-- middlename: string (nullable = true)
|-- lastname: string (nullable = true)
|-- age: string (nullable = true)
|-- gender: string (nullable = true)
|-- salary: integer (nullable = true)
```

The output of the code is displayed below the code cell. It shows the schema of the DataFrame. The schema is as follows:

firstname	middlename	lastname	age	gender	salary
James		Smith	36	M	3000
Michael	Rose		40	M	4000
Robert		Williams	42	M	4000
Maria	Anne	Jones	39	F	4000
Jen	Mary	Brown		F	-1

Command took 0.43 seconds -- by azuser924_mml.local@ihti.onmicrosoft.com at 12/29/2023, 5:51:55 PM on azuser924_Cluster

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P azdatabbiba_Hexaware azuser924_mml.local@ihl.onmicr...

python_Notebook 2023-12-29 17:42:46 Python

```
58 .withColumn("DiffInSeconds",current_timestamp().cast("long") - col("input_timestamp").cast("long")) \
59 .withColumn("DiffInMinutes",round(col("DiffInSeconds")/60)) \
60 .withColumn("DiffInHours",round(col("DiffInSeconds")/3600)) \
61 .show(truncate=False)
62
63
64 #SQL
65
66 spark.sql("select unix_timestamp('2019-07-02 12:01:19') - unix_timestamp('2019-07-01 12:01:19') DiffInSeconds").show()
67 spark.sql("select (unix_timestamp('2019-07-02 12:01:19') - unix_timestamp('2019-07-01 12:01:19'))/60 DiffInMinutes").show()
68 spark.sql("select (unix_timestamp('2019-07-02 12:01:19') - unix_timestamp('2019-07-01 12:01:19'))/3600 DiffInHours").show()
```

(15) Spark Jobs

- df: pyspark.sql.dataframe.DataFrame = [id: string, from_timestamp: string]
- df2: pyspark.sql.dataframe.DataFrame = [id: string, from_timestamp: timestamp ... 2 more fields]
- df3: pyspark.sql.dataframe.DataFrame = [id: string, input_timestamp: string]

	id	from_timestamp	end_timestamp	DiffInSeconds	DiffInMinutes
1	2019-07-01	12:01:19.111	2023-12-29 12:23:32.652	141870133	
2	2019-06-24	12:01:19.222	2023-12-29 12:23:32.652	142474934	
3	2019-11-16	16:44:55.406	2023-12-29 12:23:32.652	129929917	
4	2019-11-16	16:50:59.406	2023-12-29 12:23:32.652	129929553	

	id	from_timestamp	end_timestamp	DiffInSeconds	DiffInMinutes
1	2019-07-01	12:01:19.111	2023-12-29 12:23:33.121	141870134	2364502.0
2	2019-06-24	12:01:19.222	2023-12-29 12:23:33.121	142474934	2374582.0
3	2019-11-16	16:44:55.406	2023-12-29 12:23:33.121	129929918	2165499.0
4	2019-11-16	16:50:59.406	2023-12-29 12:23:33.121	129929554	2165493.0

	id	from_timestamp	end_timestamp	DiffInSeconds	DiffInHours
1	2019-07-01	12:01:19.111	2023-12-29 12:23:33.121	141870134	2364502.0
2	2019-06-24	12:01:19.222	2023-12-29 12:23:33.121	142474934	2374582.0
3	2019-11-16	16:44:55.406	2023-12-29 12:23:33.121	129929918	2165499.0
4	2019-11-16	16:50:59.406	2023-12-29 12:23:33.121	129929554	2165493.0

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P azdatabbiba_Hexaware azuser924_mml.local@ihl.onmicr...

python_Notebook 2023-12-29 17:42:46 Python

```
1 from pyspark.sql import SparkSession
2
3 # Create SparkSession
4 spark = SparkSession.builder \
5     .appName('SparkByExamples.com') \
6     .getOrCreate()
7
8 from pyspark.sql.functions import *
9
10 df=spark.createDataFrame([["02-03-2013"],["05-06-2023"]],["input"])
11 df.select(col("input"),to_date(col("input"),"MM-dd-yyyy").alias("date")) \
12     .show()
13
14 #SQL
15 spark.sql("select to_date('02-03-2013','MM-dd-yyyy') date").show()
16
17
```

(3) Spark Jobs

- df: pyspark.sql.dataframe.DataFrame = [input: string]

	input	date
0	02-03-2013	2013-02-03
1	05-06-2023	2023-05-06

	date
0	2013-02-03

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P azddatabbiba_Hexaware azuser924_mml.local@ihl.onmicr...

python_Notebook 2023-12-29 17:42:46 Python File Edit View Run Help Last edit was 2 minutes ago Provide feedback

```
1 import pyspark
2 from pyspark.sql import SparkSession
3 from pyspark.sql.functions import col, lit
4 from pyspark.sql.types import StructType, StructField, StringType, IntegerType
5
6 spark = SparkSession.builder.appName('SparkByExamples.com').getOrCreate()
7
8 print(spark)
9
10
```

<pyspark.sql.session.SparkSession object at 0x7f1c3254dc0>

Command took 0.14 seconds -- by azuser924_mml.local@ihl1.onmicrosoft.com at 12/29/2023, 5:56:09 PM on azuser924_Cluster

Shift+Enter to run
Shift+Ctrl+Enter to run selected text

Microsoft Azure databricks Search data, notebooks, recents, and more... CTRL + P azddatabbiba_Hexaware azuser924_mml.local@ihl.onmicr...

python_Notebook 2023-12-29 17:42:46 Python File Edit View Run Help Last edit was 3 minutes ago Provide feedback

```
20
21 # Date & Timestamp into custom format
22 df.withColumn("date_format",date_format(current_date(),"MM-dd-yyyy")) \
23   .withColumn("to_timestamp",to_timestamp(current_timestamp(),"MM-dd-yyyy HH mm ss SSS")) \
24   .show(truncate=False)
25
26 #SQL
27 spark.sql("select date_format(current_date(),'MM-dd-yyyy') as date_format ,* \
28           'to_timestamp(current_timestamp(),'MM-dd-yyyy HH mm ss SSS') as to_timestamp" ) \
29   .show(truncate=False)
```

(6) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [id: string]

id	date_format	to_timestamp
1	12-29-2023	2023-12-29 12:27:01.238

id	date_format	to_timestamp
1	12-29-2023	2023-12-29 12:27:01.601