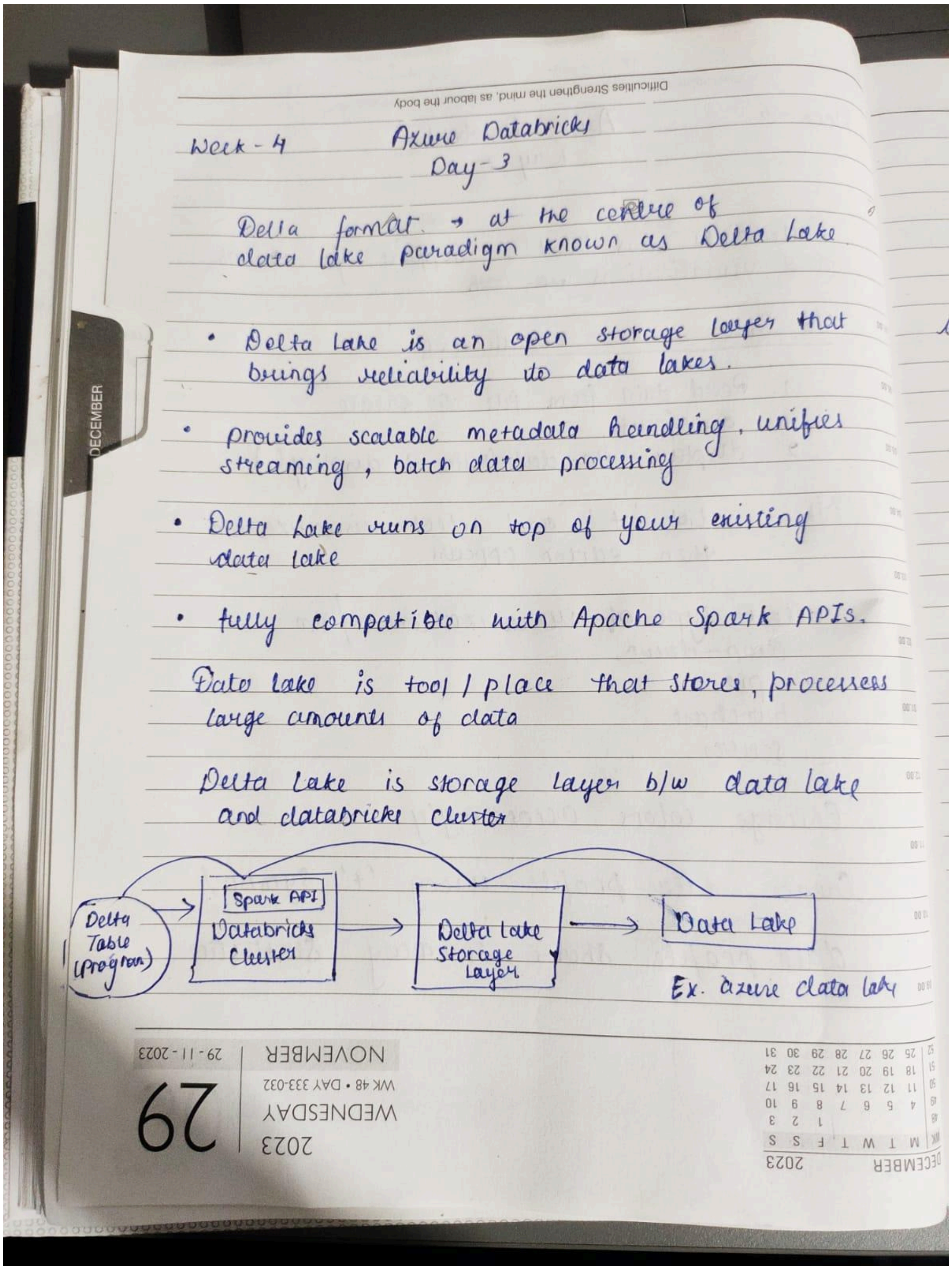
Azure Databricks Assignment - 3
Mitushi Vishwakarma

Introduction to Databricks Delta Lake



Week - 4 Azure Databricks
Day - 3

Della format → at the centre of data lake paradigm known as Delta Lake

- Delta Lake is an open storage layer that brings reliability to data lakes.
- provides scalable metadata handling, unifies streaming, batch data processing
- Delta Lake runs on top of your existing data lake
- fully compatible with Apache Spark APIs.

Data lake is tool / place that stores, processes large amounts of data

Delta Lake is storage layer b/w data lake and databricks cluster

How to Create a Delta Table.

Step 1 : Uploading data to DBFS
2 : data in delta format.
:

Create a Table :-
Write dataframe in delta format.

Read a Table :-
read data by specifying the path.
SQL delta.`/tmp/delta-table`;
python .format( ). load( )

Update
mode = overwrite

.format( ) . mode( ) . Save( )

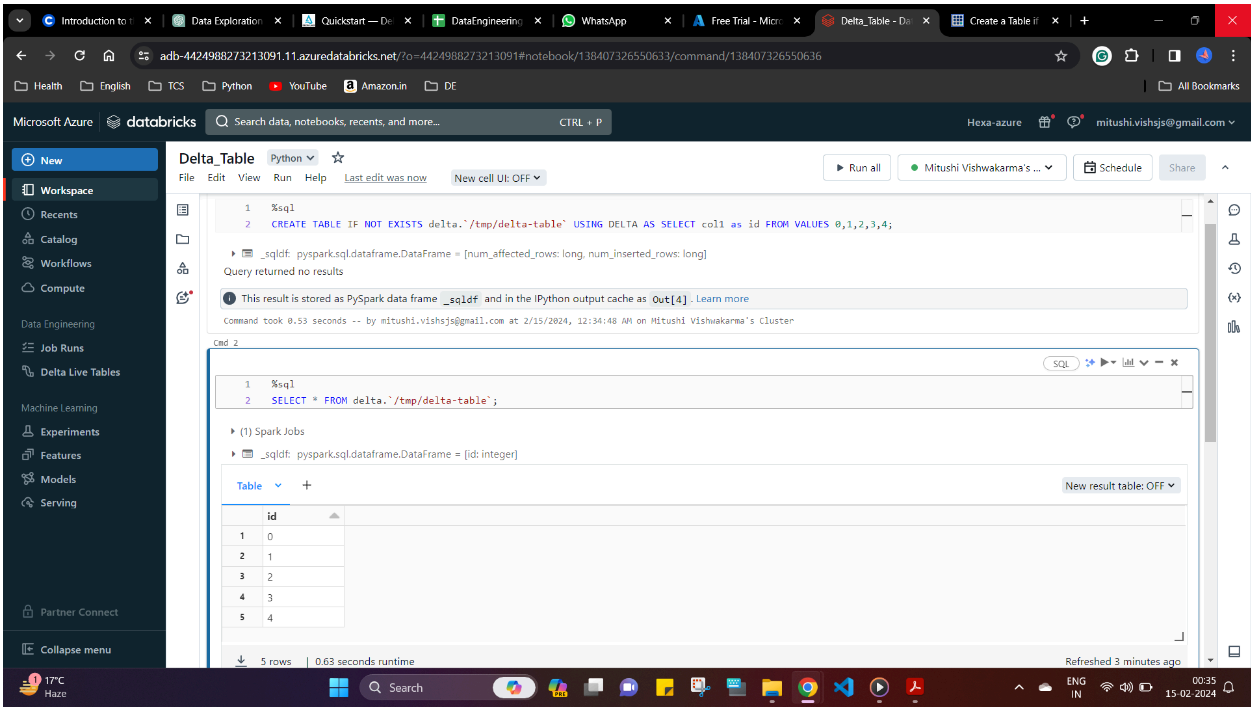
Reading Older version.
.option( ). Load(,

NOVEMBER 2023

WK	M	T	W	T	F	S	S
44			1	2	3	4	5
45	6	7	8	9	10	11	12
46	13	14	15	16	17	18	19
47	20	21	22	23	24	25	26
48	27	28	29	30			

Practice :

Creating delta table and reading data in sql



Delta_Table

```
%sql
CREATE TABLE IF NOT EXISTS delta.`/tmp/delta-table` USING DELTA AS SELECT col1 as id FROM VALUES 0,1,2,3,4;
```

_sqldf: pyspark.sql.dataframe.DataFrame = [num_affected_rows: long, num_inserted_rows: long]

Query returned no results

This result is stored as PySpark data frame `_sqldf` and in the IPython output cache as `Out[4]`. Learn more

```
Command took 0.53 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:34:48 AM on Mitushi Vishwakarma's Cluster
```

Cmd 2

```
%sql
SELECT * FROM delta.`/tmp/delta-table`;
```

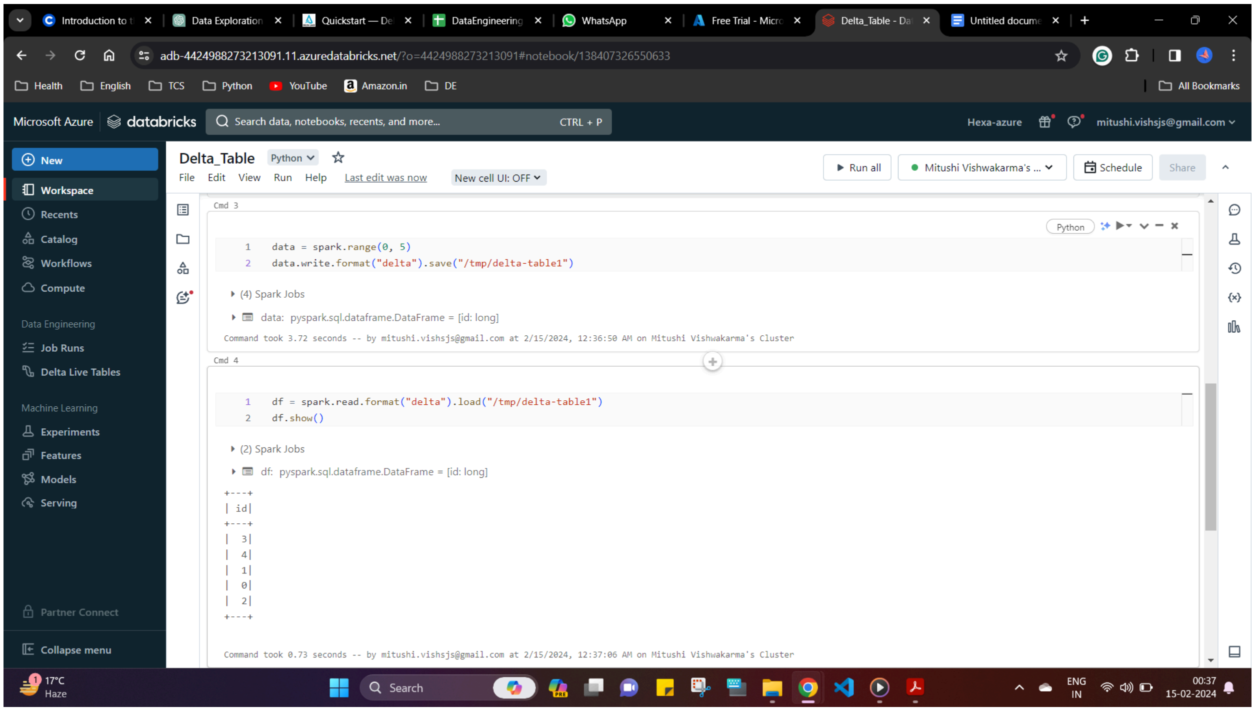
(1) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [id: integer]

	id
1	0
2	1
3	2
4	3
5	4

5 rows | 0.63 seconds runtime   Refreshed 3 minutes ago

Creating delta table and reading data in python



Delta_Table

Cmd 3

```
data = spark.range(0, 5)
data.write.format("delta").save("/tmp/delta-table1")
```

(4) Spark Jobs

data: pyspark.sql.dataframe.DataFrame = [id: long]

```
Command took 3.72 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:36:50 AM on Mitushi Vishwakarma's Cluster
```

Cmd 4

```
df = spark.read.format("delta").load("/tmp/delta-table1")
df.show()
```

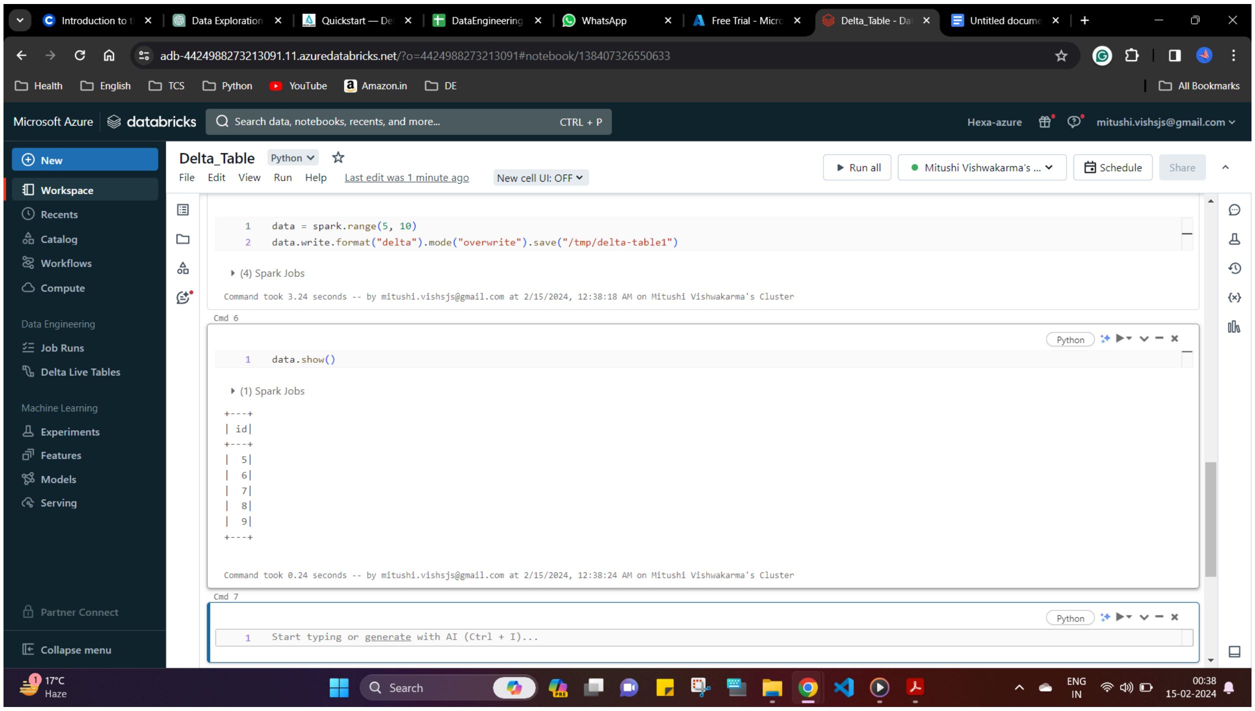
(2) Spark Jobs

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

```
+---+
| id|
+---+
|  3|
|  4|
|  1|
|  0|
|  2|
+---+
```

```
Command took 0.73 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:37:06 AM on Mitushi Vishwakarma's Cluster
```

Updating data in delta table in python



```
data = spark.range(5, 10)
data.write.format("delta").mode("overwrite").save("/tmp/delta-table1")
```

```
(4) Spark Jobs
Command took 3.24 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:38:18 AM on Mitushi Vishwakarma's Cluster
```

Cmd 6

```
data.show()
```

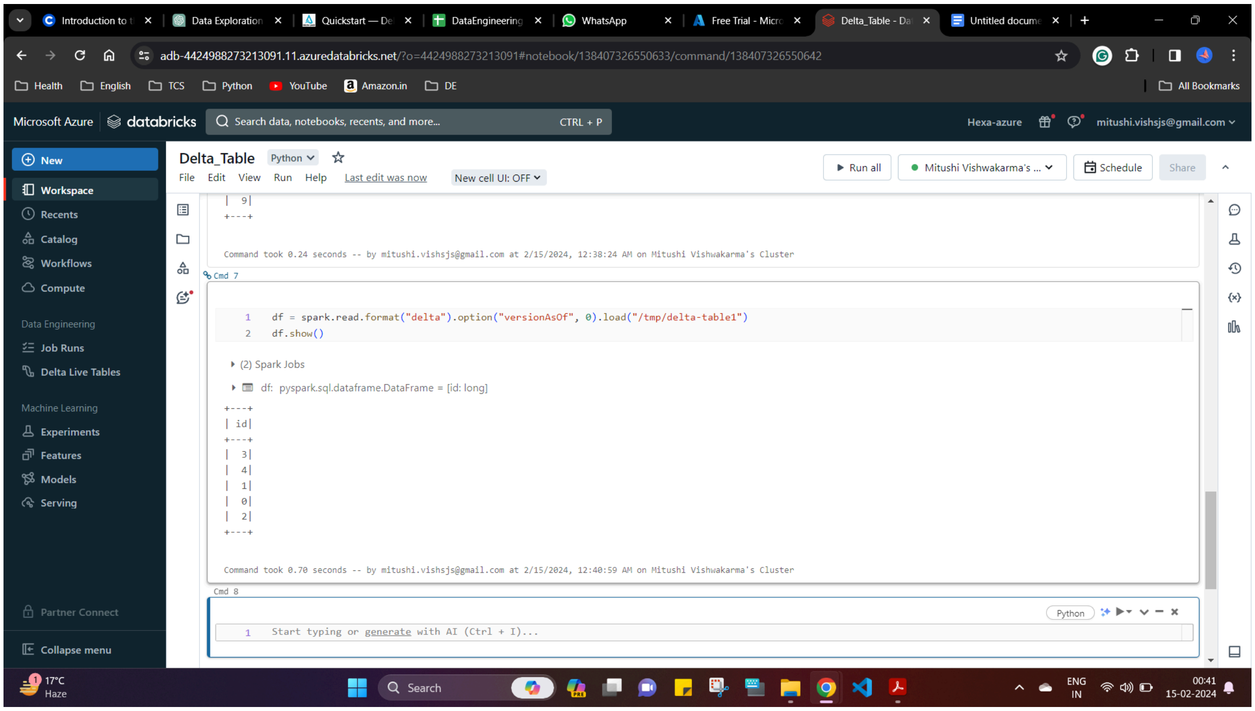
```
(1) Spark Jobs
+---+
| id|
+---+
|  5|
|  6|
|  7|
|  8|
|  9|
+---+

Command took 0.24 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:38:24 AM on Mitushi Vishwakarma's Cluster
```

Cmd 7

```
Start typing or generate with AI (Ctrl + I)...
```

Read older versions of data using time travel



```
|  9|
+---+

Command took 0.24 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:38:24 AM on Mitushi Vishwakarma's Cluster
```

Cmd 7

```
df = spark.read.format("delta").option("versionAsOf", 0).load("/tmp/delta-table1")
df.show()
```

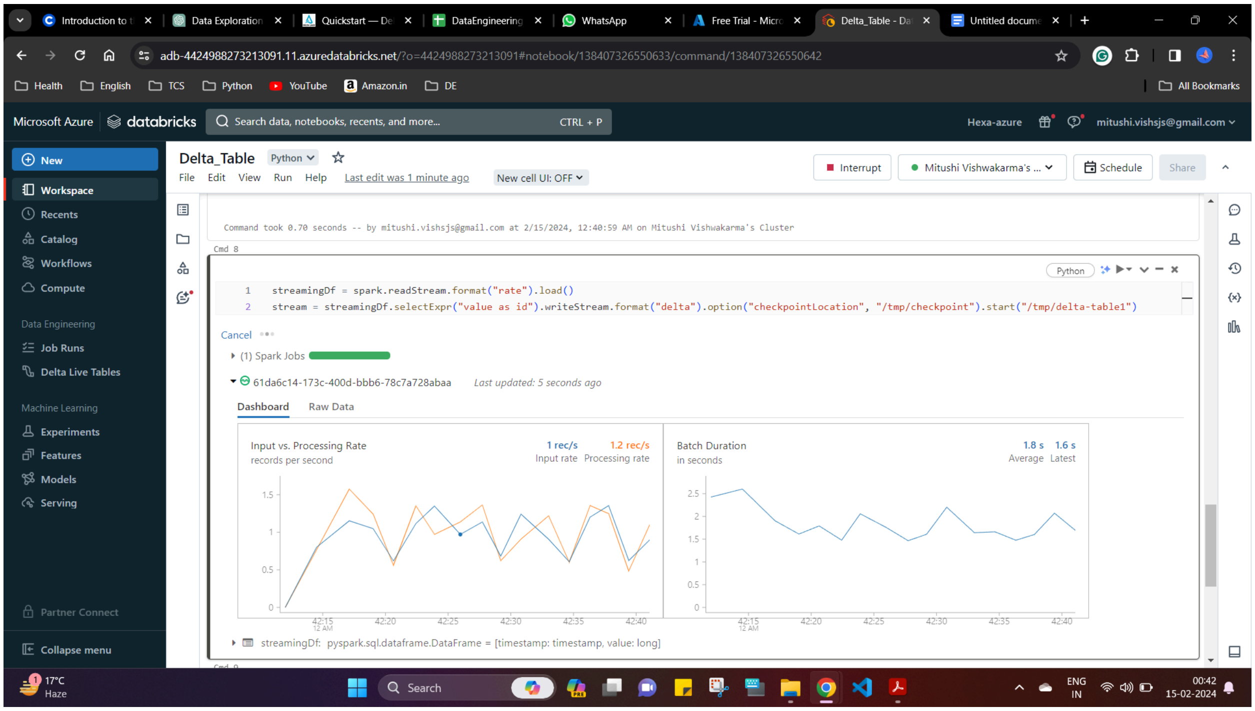
```
(2) Spark Jobs
df: pyspark.sql.dataframe.DataFrame = [id: long]
+---+
| id|
+---+
|  3|
|  4|
|  1|
|  0|
|  2|
+---+

Command took 0.70 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:40:59 AM on Mitushi Vishwakarma's Cluster
```

Cmd 8

```
Start typing or generate with AI (Ctrl + I)...
```

Write a stream of data to a table



```
Command took 0.70 seconds -- by mitushi.vishsjs@gmail.com at 2/15/2024, 12:40:59 AM on Mitushi Vishwakarma's Cluster
```

Cmd 8

```python
streamingDf = spark.readStream.format("rate").load()
stream = streamingDf.selectExpr("value as id").writeStream.format("delta").option("checkpointLocation", "/tmp/checkpoint").start("/tmp/delta-table1")
```

Cancel

(1) Spark Jobs

61da6c14-173c-400d-bbb6-78c7a728abaa   *Last updated: 5 seconds ago*

Dashboard   Raw Data

Input vs. Processing Rate — records per second — 1 rec/s Input rate — 1.2 rec/s Processing rate

Batch Duration — in seconds — 1.8 s Average — 1.6 s Latest

streamingDf: pyspark.sql.dataframe.DataFrame = [timestamp: timestamp, value: long]