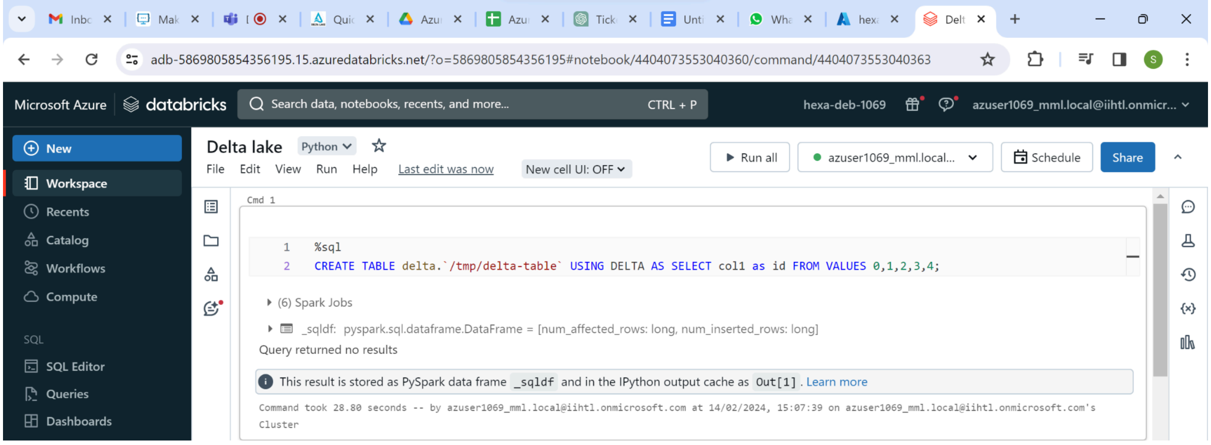
NAME : AKULA SHARATH CHANDRA
BATCH : DATA ENGINEERING
DATE : 14-02-2024
TOPICS :

1. **Creating a delta table and passing values**
2. **Creating and visualising the stream data Create a table.**

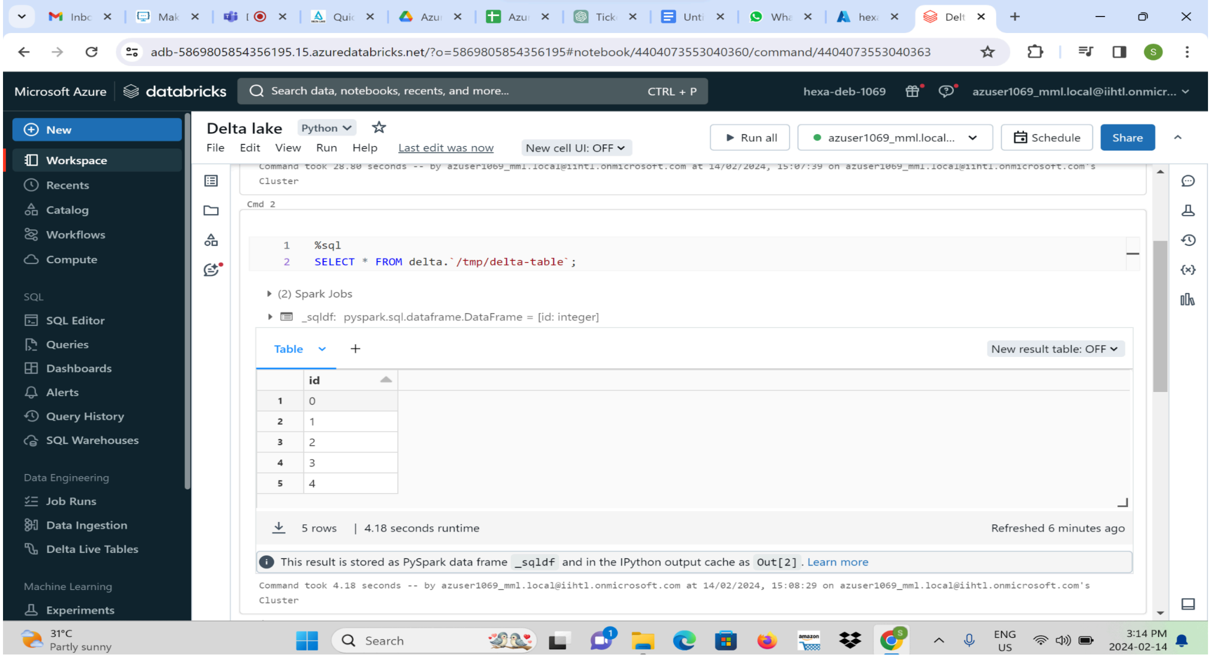
To create a Delta table, write a DataFrame out in the delta format. You can use existing Spark SQL code and change the format from parquet, csv, json, and so on, to delta.



```
%sql
CREATE TABLE delta.`/tmp/delta-table` USING DELTA AS SELECT col1 as id FROM VALUES 0,1,2,3,4;
```

```
(6) Spark Jobs
_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[1]. Learn more
Command took 28.80 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:07:39 on azuser1069_mml.local@iihtl.onmicrosoft.com's Cluster
```

Read data You read data in your Delta table by specifying the path to the files: "/tmp/delta-table":



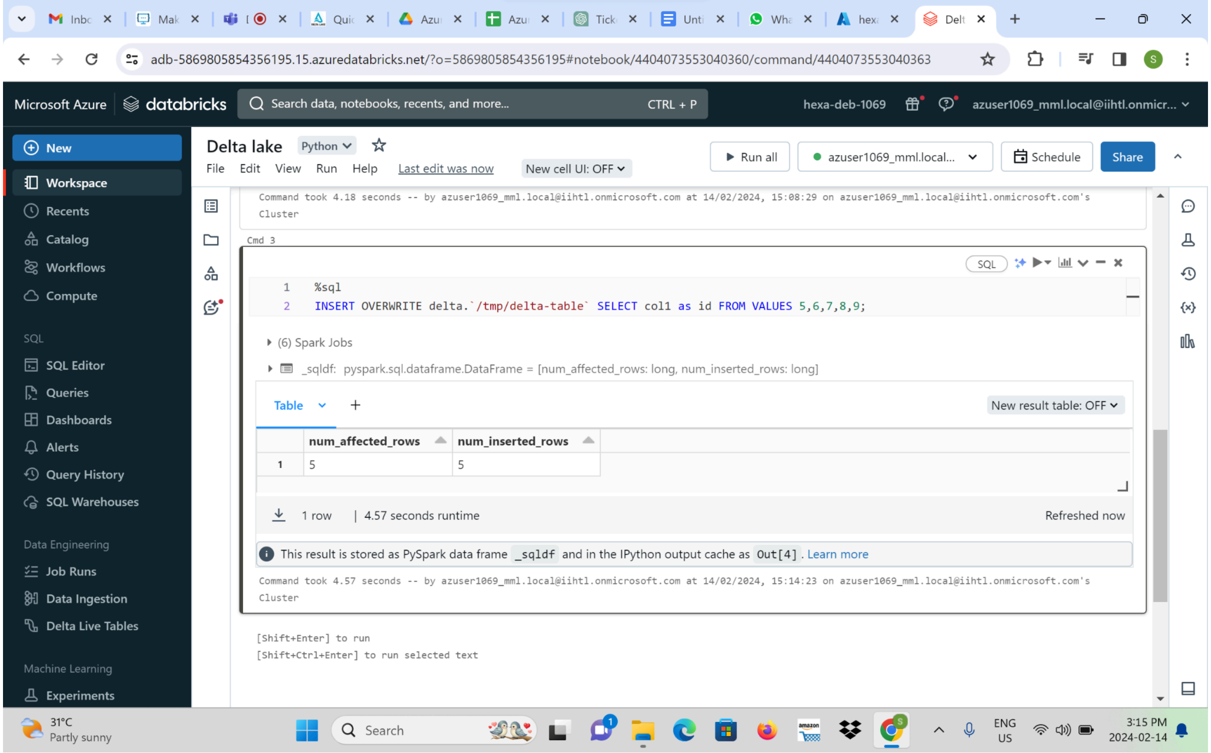
```
%sql
SELECT * FROM delta.`/tmp/delta-table`;
```

```
(2) Spark Jobs
_sqldf:  pyspark.sql.dataframe.DataFrame = [id: integer]
```

	id
1	0
2	1
3	2
4	3
5	4

```
5 rows  | 4.18 seconds runtime                                    Refreshed 6 minutes ago
This result is stored as PySpark data frame _sqldf and in the IPython output cache as Out[2]. Learn more
Command took 4.18 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:08:29 on azuser1069_mml.local@iihtl.onmicrosoft.com's Cluster
```

Update table data Delta Lake supports several operations to modify tables using standard DataFrame APIs. This example runs a batch job to overwrite the data in the table:



```
Command took 4.18 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:08:29 on azuser1069_mml.local@iihtl.onmicrosoft.com's
Cluster
```

```
%sql
INSERT OVERWRITE delta.`/tmp/delta-table` SELECT col1 as id FROM VALUES 5,6,7,8,9;
```

```
(6) Spark Jobs
_sqldf:  pyspark.sql.dataframe.DataFrame = [num_affected_rows: long, num_inserted_rows: long]
```

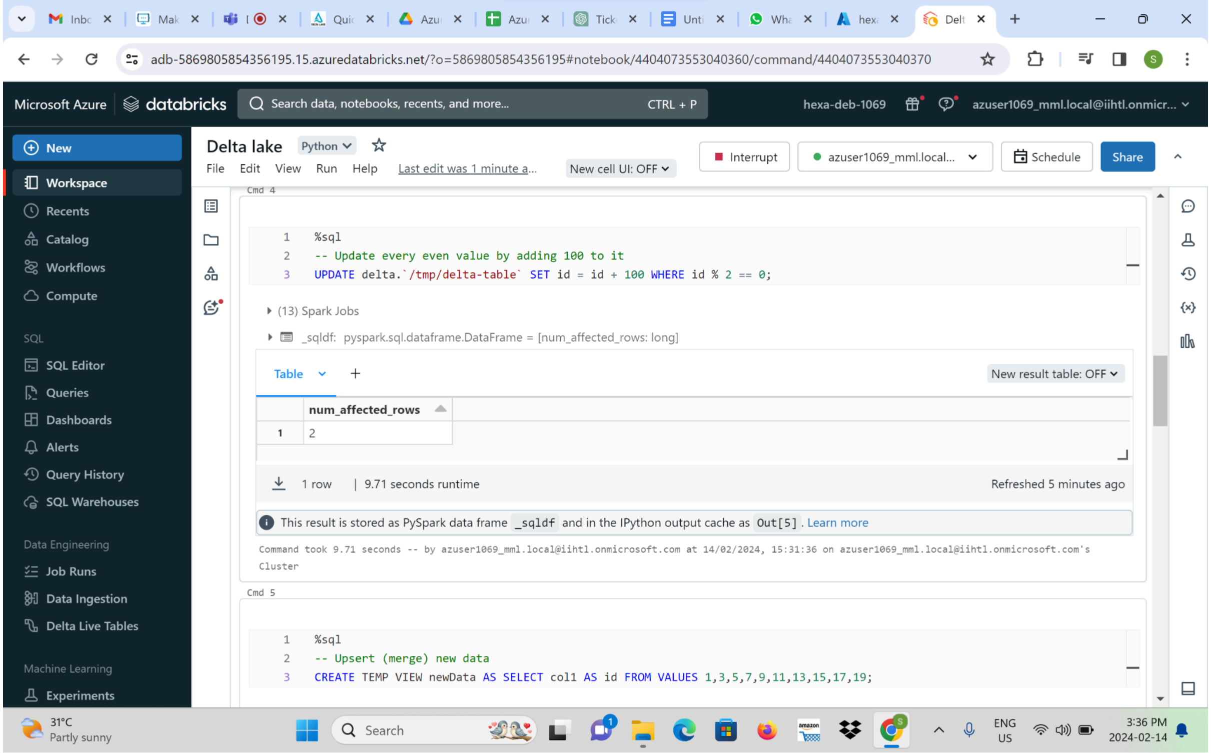
	num_affected_rows	num_inserted_rows
1	5	5

1 row | 4.57 seconds runtime Refreshed now

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

```
Command took 4.57 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:14:23 on azuser1069_mml.local@iihtl.onmicrosoft.com's
Cluster

[Shift+Enter] to run
[Shift+Ctrl+Enter] to run selected text
```

Conditional update without overwrite Delta Lake provides programmatic APIs to conditional update, delete, and merge (upsert) data into tables. Here are a few examples.



```
%sql
-- Update every even value by adding 100 to it
UPDATE delta.`/tmp/delta-table` SET id = id + 100 WHERE id % 2 == 0;
```

(13) Spark Jobs

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

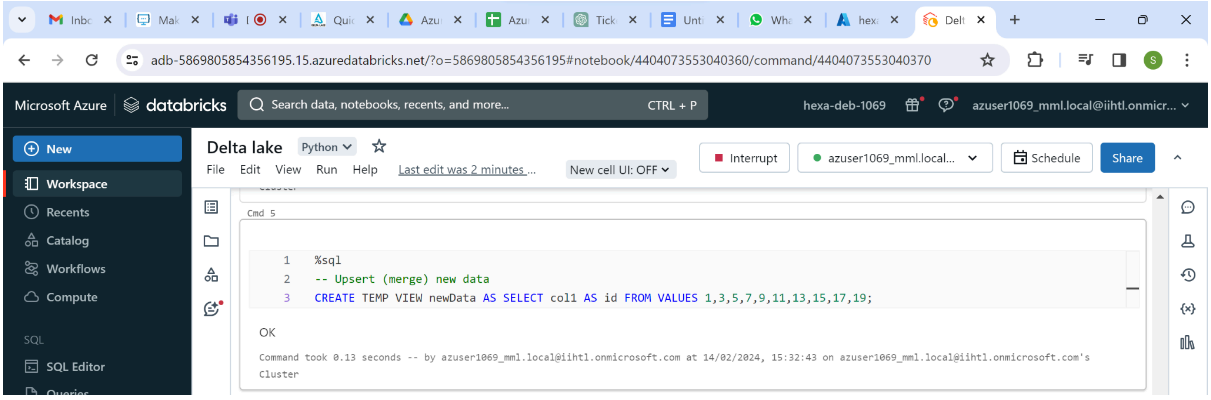
	num_affected_rows
1	2

1 row | 9.71 seconds runtime   Refreshed 5 minutes ago

This result is stored as PySpark data frame `_sqldf` and in the IPython output cache as `Out[5]`. Learn more

```
Command took 9.71 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:31:36 on azuser1069_mml.local@iihtl.onmicrosoft.com's Cluster
```

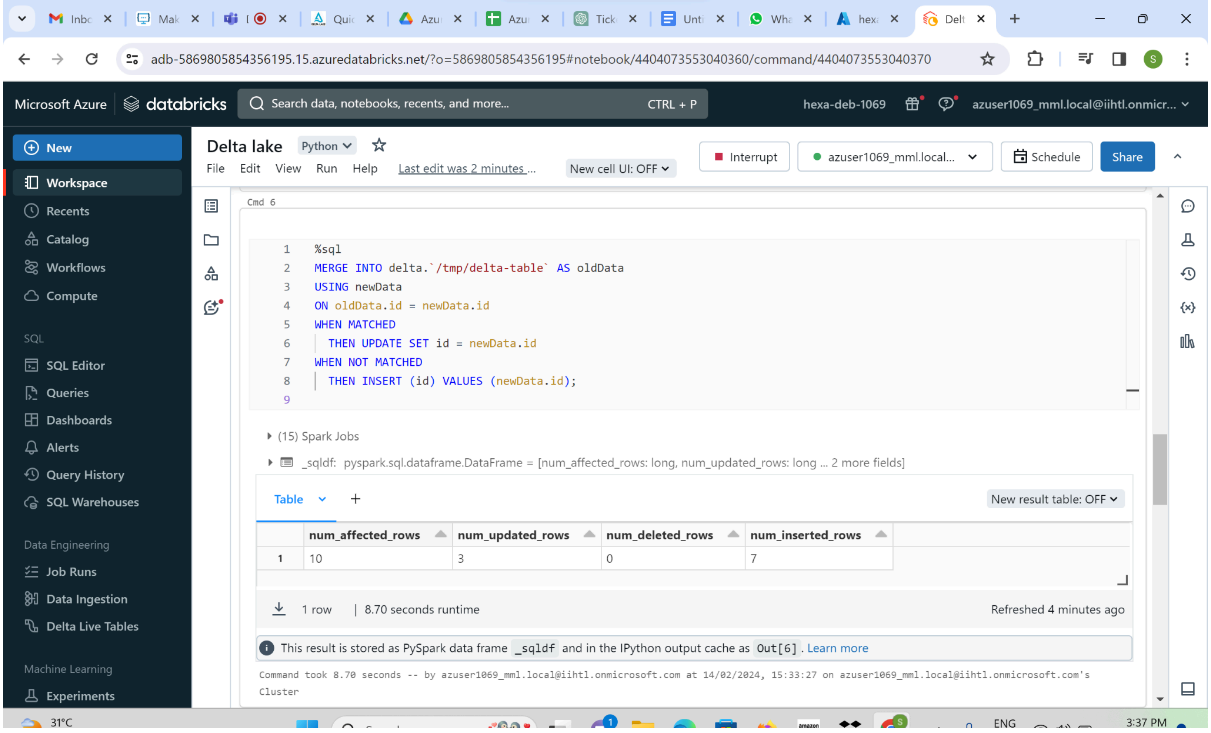
```
%sql
-- Upsert (merge) new data
CREATE TEMP VIEW newData AS SELECT col1 AS id FROM VALUES 1,3,5,7,9,11,13,15,17,19;
```


adb-5869805854356195.15.azuredatabricks.net/?o=5869805854356195#notebook/4404073553040360/command/4404073553040370
Microsoft Azure
databricks
Search data, notebooks, recents, and more...
CTRL + P
hexa-deb-1069
azuser1069_mml.local@iihtl.onmicr...
New
Workspace
Recents
Catalog
Workflows
Compute
SQL
SQL Editor
Delta lake
Python
File
Edit
View
Run
Help
Last edit was 2 minutes ...
New cell UI: OFF
Interrupt
azuser1069_mml.local...
Schedule
Share
Cmd 5

```
%sql
-- Upsert (merge) new data
CREATE TEMP VIEW newData AS SELECT col1 AS id FROM VALUES 1,3,5,7,9,11,13,15,17,19;
```

OK

```
Command took 0.13 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:32:43 on azuser1069_mml.local@iihtl.onmicrosoft.com's Cluster
```



```
%sql
MERGE INTO delta.`/tmp/delta-table` AS oldData
USING newData
ON oldData.id = newData.id
WHEN MATCHED
  THEN UPDATE SET id = newData.id
WHEN NOT MATCHED
  THEN INSERT (id) VALUES (newData.id);
```

(15) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [num_affected_rows: long, num_updated_rows: long ... 2 more fields]

	num_affected_rows	num_updated_rows	num_deleted_rows	num_inserted_rows
1	10	3	0	7

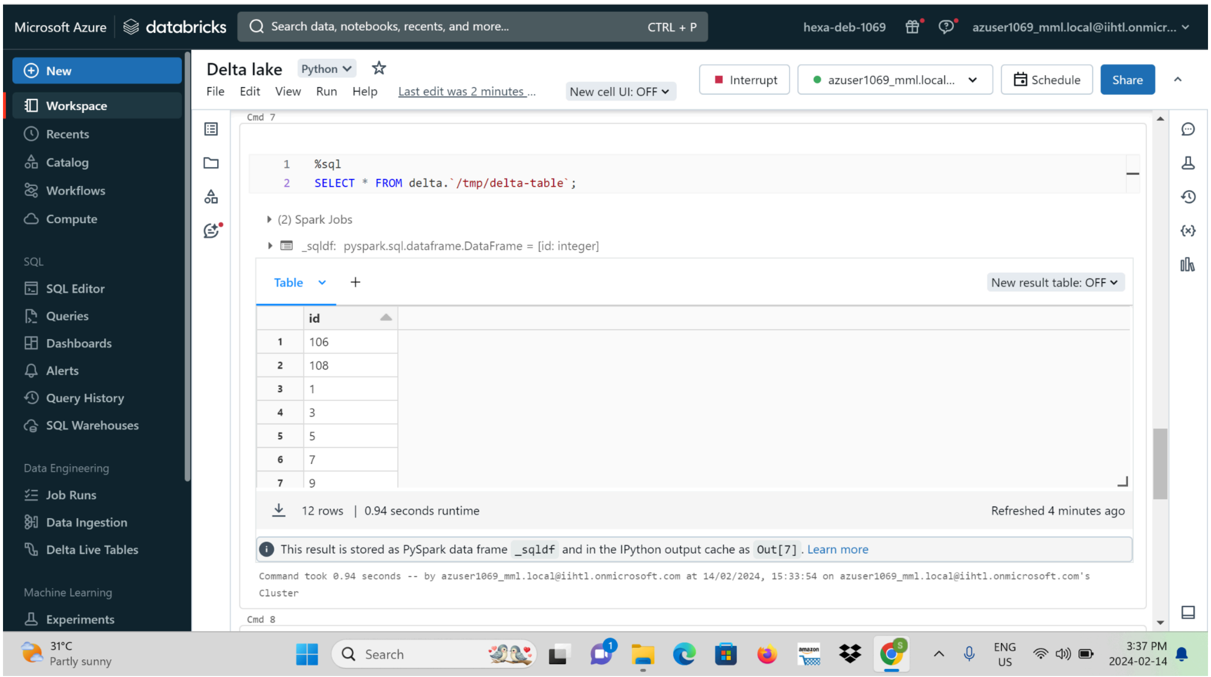
1 row | 8.70 seconds runtime

Refreshed 4 minutes ago

This result is stored as PySpark data frame `_sqldf` and in the IPython output cache as `Out[6]`. Learn more

```
Command took 8.70 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:33:27 on azuser1069_mml.local@iihtl.onmicrosoft.com's Cluster
```

You can query previous snapshots of your Delta table by using time travel. If you want to access the data that you overwrote, you can query a snapshot of the table before you overwrote the first set of data using the versionAsOf option.



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

(2) Spark Jobs

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

	id
1	106
2	108
3	1
4	3
5	5
6	7
7	9

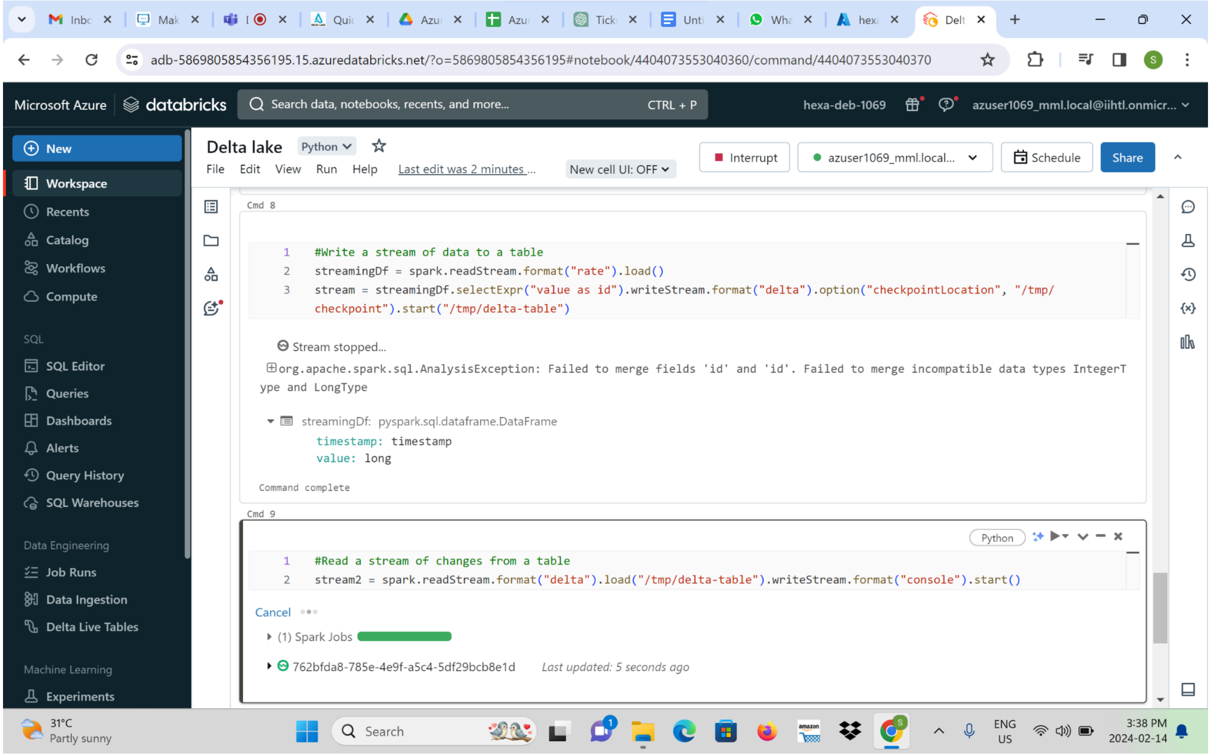
12 rows | 0.94 seconds runtime

Refreshed 4 minutes ago

This result is stored as PySpark data frame `_sqldf` and in the IPython output cache as `Out[7]`. Learn more

```
Command took 0.94 seconds -- by azuser1069_mml.local@iihtl.onmicrosoft.com at 14/02/2024, 15:33:54 on azuser1069_mml.local@iihtl.onmicrosoft.com's Cluster
```

Write a stream of data to a table You can also write to a Delta table using Structured Streaming. The Delta Lake transaction log guarantees exactly-once processing, even when there are other streams or batch queries running concurrently against the table. By default, streams run in append mode, which adds new records to the table:



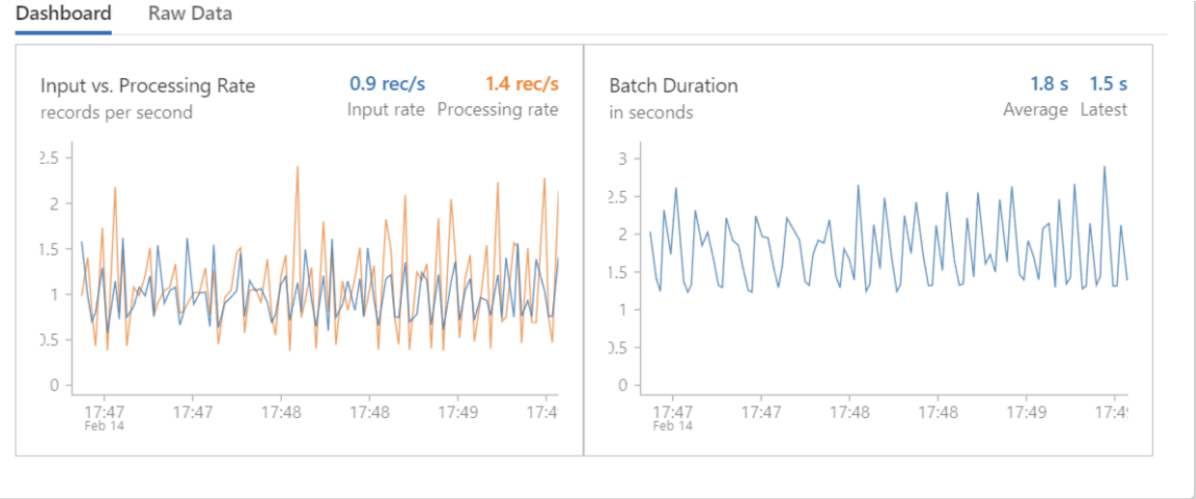
adb-5869805854356195.15.azuredatabricks.net/?o=5869805854356195#notebook/4404073553040360/command/4404073553040370

```
#Write a stream of data to a table
streamingDf = spark.readStream.format("rate").load()
stream = streamingDf.selectExpr("value as id").writeStream.format("delta").option("checkpointLocation", "/tmp/
checkpoint").start("/tmp/delta-table")
```

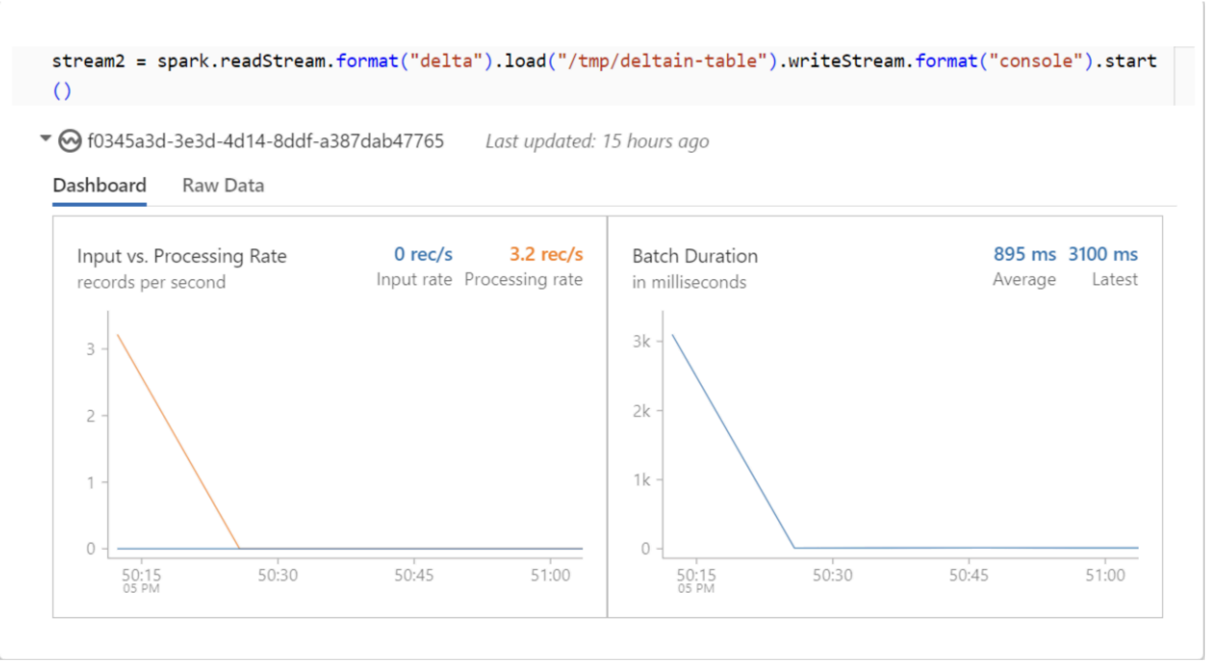
```
Stream stopped...
org.apache.spark.sql.AnalysisException: Failed to merge fields 'id' and 'id'. Failed to merge incompatible data types IntegerT
ype and LongType

streamingDf:  pyspark.sql.dataframe.DataFrame
    timestamp: timestamp
    value: long

Command complete
```

```
#Read a stream of changes from a table
stream2 = spark.readStream.format("delta").load("/tmp/delta-table").writeStream.format("console").start()
```

```
762bfda8-785e-4e9f-a5c4-5df29bcb8e1d    Last updated: 5 seconds ago
```



Read a stream of changes from a table While the stream is writing to the Delta table, you can also read from that table as streaming source. For example, you can start another streaming query that prints all the changes made to the Delta table. You can specify which version Structured Streaming should start from by providing the startingVersion or startingTimestamp option to get changes from that point onwards. See Structured Streaming for details.



```
stream2 = spark.readStream.format("delta").load("/tmp/deltain-table").writeStream.format("console").start()
```

f0345a3d-3e3d-4d14-8ddf-a387dab47765 *Last updated: 15 hours ago*