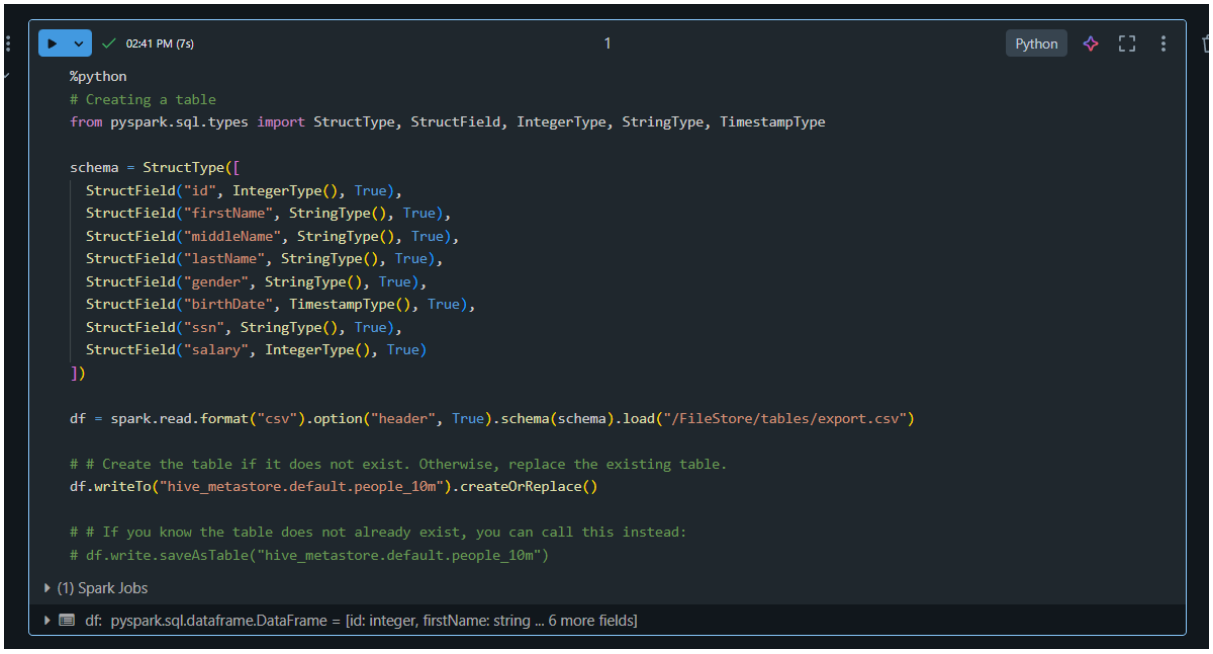
ASSIGNMENT – 4 (Azure Databricks)

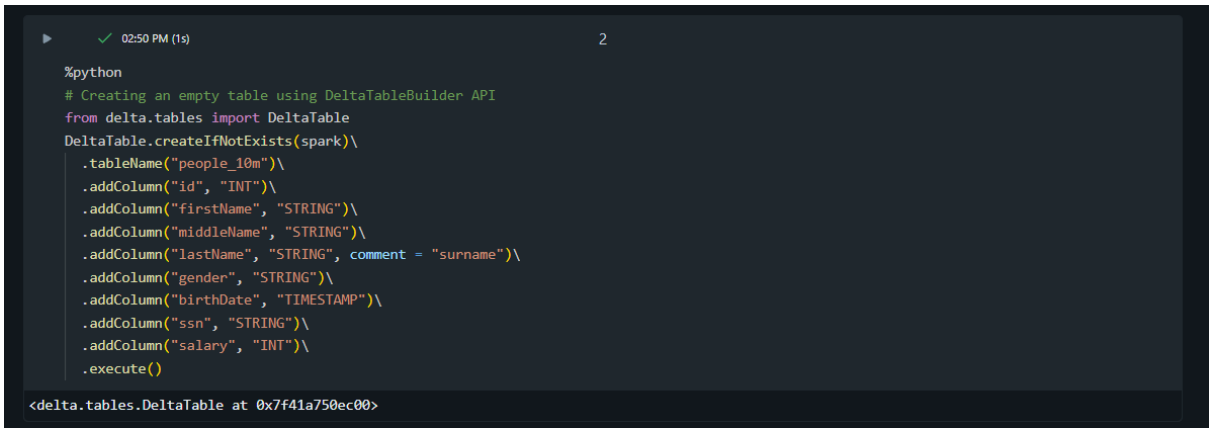
TRAINEE NAME: Swathi Baskaran

Creating a Table



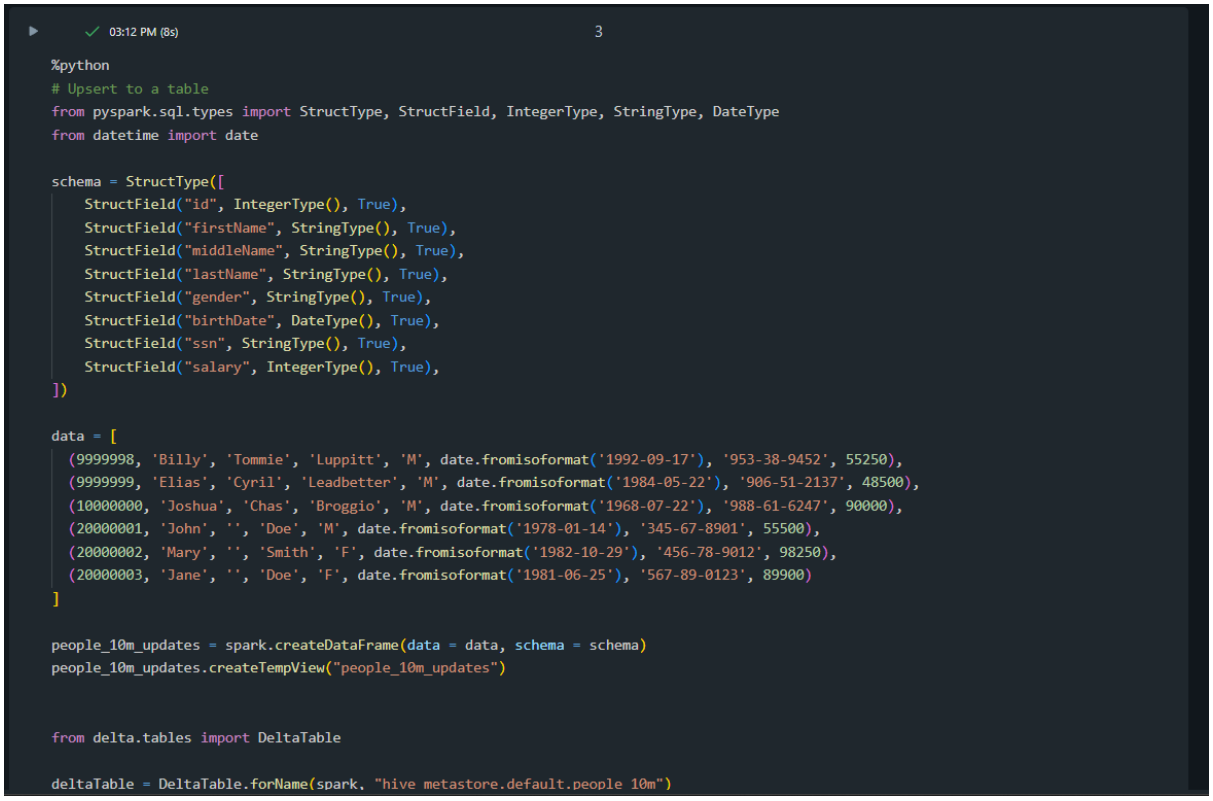
```
02:41 PM (7s)                                                    1                                                    Python

%python
# Creating a table
from pyspark.sql.types import StructType, StructField, IntegerType, StringType, TimestampType

schema = StructType([
  StructField("id", IntegerType(), True),
  StructField("firstName", StringType(), True),
  StructField("middleName", StringType(), True),
  StructField("lastName", StringType(), True),
  StructField("gender", StringType(), True),
  StructField("birthDate", TimestampType(), True),
  StructField("ssn", StringType(), True),
  StructField("salary", IntegerType(), True)
])

df = spark.read.format("csv").option("header", True).schema(schema).load("/FileStore/tables/export.csv")

# # Create the table if it does not exist. Otherwise, replace the existing table.
df.writeTo("hive_metastore.default.people_10m").createOrReplace()

# # If you know the table does not already exist, you can call this instead:
# df.write.saveAsTable("hive_metastore.default.people_10m")

▸ (1) Spark Jobs

▸ df:  pyspark.sql.dataframe.DataFrame = [id: integer, firstName: string ... 6 more fields]
```

Creating an empty table using DeltaTableBuilderAPI



```
02:50 PM (1s)                                                    2

%python
# Creating an empty table using DeltaTableBuilder API
from delta.tables import DeltaTable
DeltaTable.createIfNotExists(spark)\
  .tableName("people_10m")\
  .addColumn("id", "INT")\
  .addColumn("firstName", "STRING")\
  .addColumn("middleName", "STRING")\
  .addColumn("lastName", "STRING", comment = "surname")\
  .addColumn("gender", "STRING")\
  .addColumn("birthDate", "TIMESTAMP")\
  .addColumn("ssn", "STRING")\
  .addColumn("salary", "INT")\
  .execute()

<delta.tables.DeltaTable at 0x7f41a750ec00>
```

Upsert to a Table



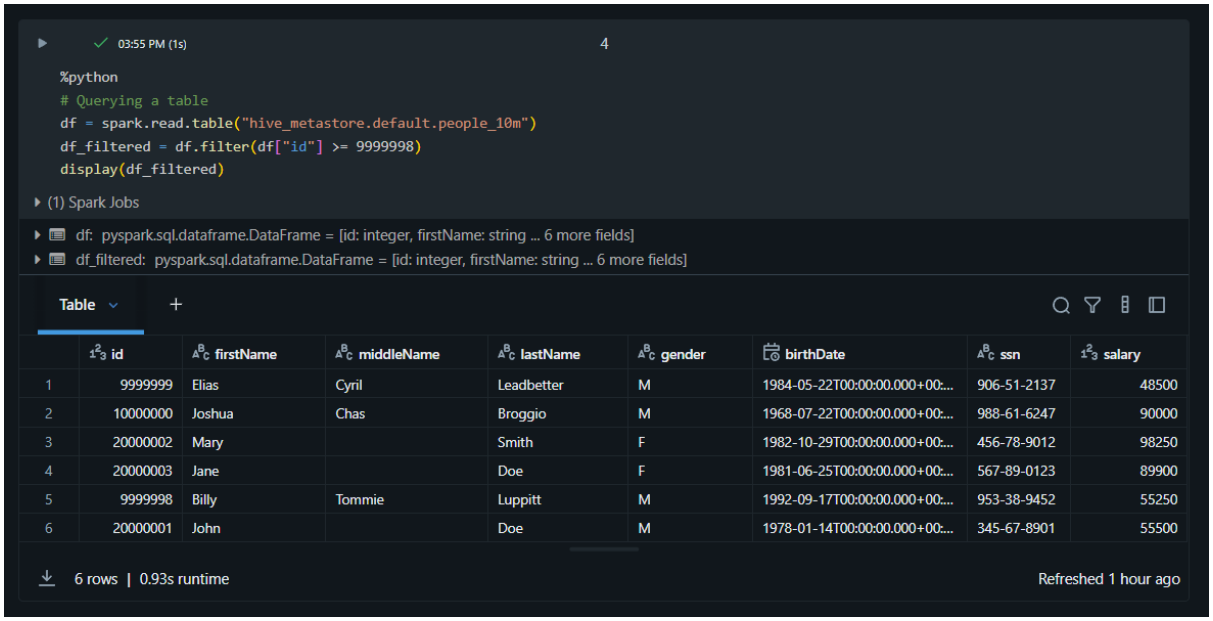
```
%python
# Upsert to a table
from pyspark.sql.types import StructType, StructField, IntegerType, StringType, DateType
from datetime import date

schema = StructType([
    StructField("id", IntegerType(), True),
    StructField("firstName", StringType(), True),
    StructField("middleName", StringType(), True),
    StructField("lastName", StringType(), True),
    StructField("gender", StringType(), True),
    StructField("birthDate", DateType(), True),
    StructField("ssn", StringType(), True),
    StructField("salary", IntegerType(), True),
])

data = [
  (9999998, 'Billy', 'Tommie', 'Luppitt', 'M', date.fromisoformat('1992-09-17'), '953-38-9452', 55250),
  (9999999, 'Elias', 'Cyril', 'Leadbetter', 'M', date.fromisoformat('1984-05-22'), '906-51-2137', 48500),
  (10000000, 'Joshua', 'Chas', 'Broggio', 'M', date.fromisoformat('1968-07-22'), '988-61-6247', 90000),
  (20000001, 'John', '', 'Doe', 'M', date.fromisoformat('1978-01-14'), '345-67-8901', 55500),
  (20000002, 'Mary', '', 'Smith', 'F', date.fromisoformat('1982-10-29'), '456-78-9012', 98250),
  (20000003, 'Jane', '', 'Doe', 'F', date.fromisoformat('1981-06-25'), '567-89-0123', 89900)
]

people_10m_updates = spark.createDataFrame(data = data, schema = schema)
people_10m_updates.createTempView("people_10m_updates")


from delta.tables import DeltaTable

deltaTable = DeltaTable.forName(spark, "hive_metastore.default.people_10m")
```

```
from delta.tables import DeltaTable

deltaTable = DeltaTable.forName(spark, "hive_metastore.default.people_10m")

(deltaTable.alias("people_10m")
  .merge(
    people_10m_updates.alias("people_10m_updates"),
    "people_10m.id = people_10m_updates.id")
  .whenMatchedUpdateAll()
  .whenNotMatchedInsertAll()
  .execute()
)
```

(5) Spark Jobs

people_10m_updates: pyspark.sql.dataframe.DataFrame = [id: integer, firstName: string ... 6 more fields]

Querying a Table



```
%python
# Querying a table
df = spark.read.table("hive_metastore.default.people_10m")
df_filtered = df.filter(df["id"] >= 9999998)
display(df_filtered)
```

03:55 PM (1s) 4

(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [id: integer, firstName: string ... 6 more fields]

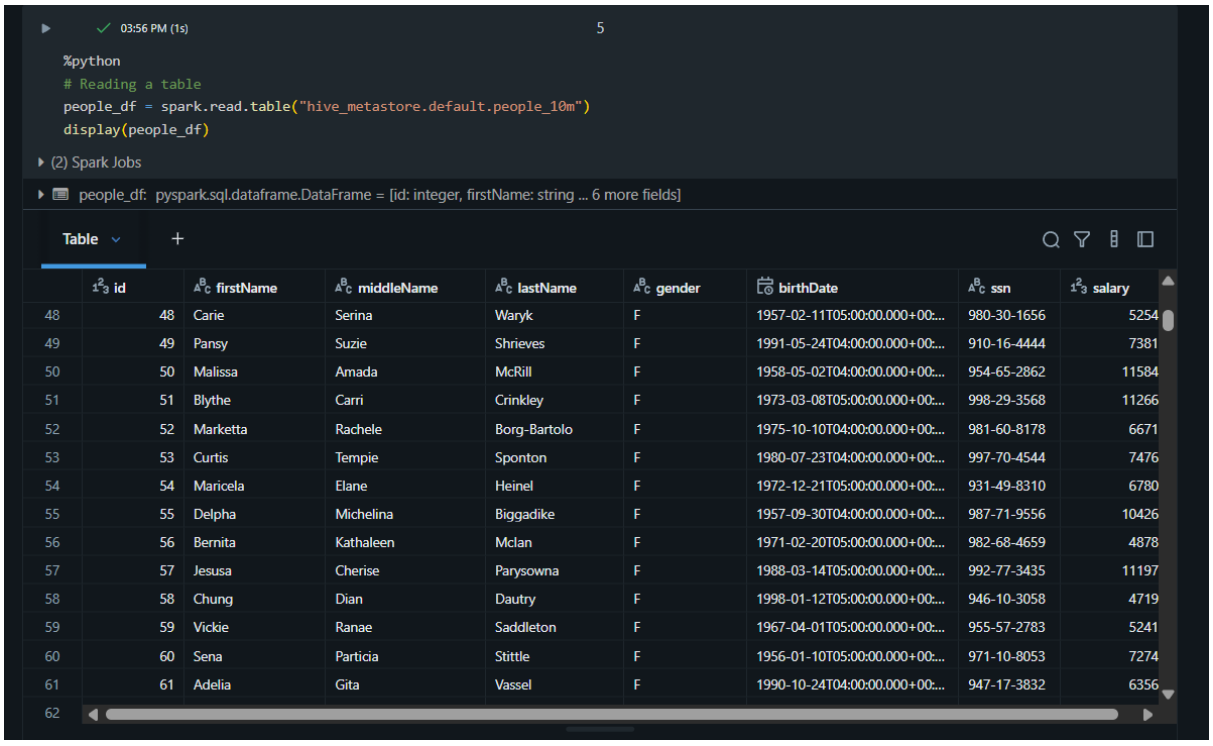
df_filtered: pyspark.sql.dataframe.DataFrame = [id: integer, firstName: string ... 6 more fields]

Table +

	id	firstName	middleName	lastName	gender	birthDate	ssn	salary
1	9999999	Elias	Cyril	Leadbetter	M	1984-05-22T00:00:00.000+00:...	906-51-2137	48500
2	10000000	Joshua	Chas	Broggio	M	1968-07-22T00:00:00.000+00:...	988-61-6247	90000
3	20000002	Mary		Smith	F	1982-10-29T00:00:00.000+00:...	456-78-9012	98250
4	20000003	Jane		Doe	F	1981-06-25T00:00:00.000+00:...	567-89-0123	89900
5	9999998	Billy	Tommie	Luppitt	M	1992-09-17T00:00:00.000+00:...	953-38-9452	55250
6	20000001	John		Doe	M	1978-01-14T00:00:00.000+00:...	345-67-8901	55500

6 rows | 0.93s runtime Refreshed 1 hour ago

Reading a Table



```
%python
# Reading a table
people_df = spark.read.table("hive_metastore.default.people_10m")
display(people_df)
```

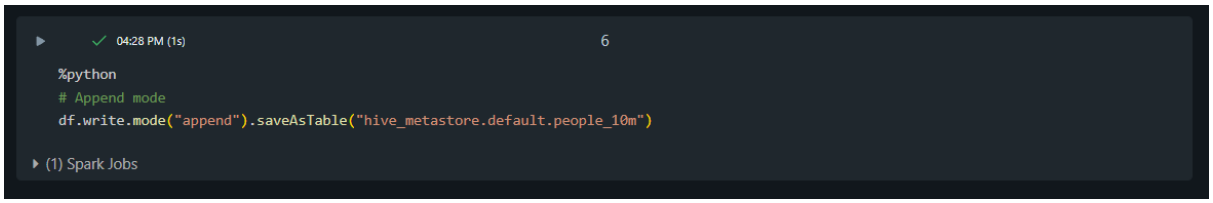
03:56 PM (1s) 5

(2) Spark Jobs

people_df: pyspark.sql.dataframe.DataFrame = [id: integer, firstName: string ... 6 more fields]

Table +

	id	firstName	middleName	lastName	gender	birthDate	ssn	salary
48	48	Carie	Serina	Waryk	F	1957-02-11T05:00:00.000+00:...	980-30-1656	5254
49	49	Pansy	Suzie	Shrieves	F	1991-05-24T04:00:00.000+00:...	910-16-4444	7381
50	50	Malissa	Amada	McRill	F	1958-05-02T04:00:00.000+00:...	954-65-2862	11584
51	51	Blythe	Carri	Crinkley	F	1973-03-08T05:00:00.000+00:...	998-29-3568	11266
52	52	Marketta	Rachele	Borg-Bartolo	F	1975-10-10T04:00:00.000+00:...	981-60-8178	6671
53	53	Curtis	Tempie	Sponton	F	1980-07-23T04:00:00.000+00:...	997-70-4544	7476
54	54	Maricela	Elane	Heinel	F	1972-12-21T05:00:00.000+00:...	931-49-8310	6780
55	55	Delpha	Michelina	Biggadike	F	1957-09-30T04:00:00.000+00:...	987-71-9556	10426
56	56	Bernita	Kathaleen	Mclan	F	1971-02-20T05:00:00.000+00:...	982-68-4659	4878
57	57	Jesusa	Cherise	Parysowna	F	1988-03-14T05:00:00.000+00:...	992-77-3435	11197
58	58	Chung	Dian	Dautry	F	1998-01-12T05:00:00.000+00:...	946-10-3058	4719
59	59	Vickie	Ranae	Saddleton	F	1967-04-01T05:00:00.000+00:...	955-57-2783	5241
60	60	Sena	Particia	Stittle	F	1956-01-10T05:00:00.000+00:...	971-10-8053	7274
61	61	Adelia	Gita	Vassel	F	1990-10-24T04:00:00.000+00:...	947-17-3832	6356
62								



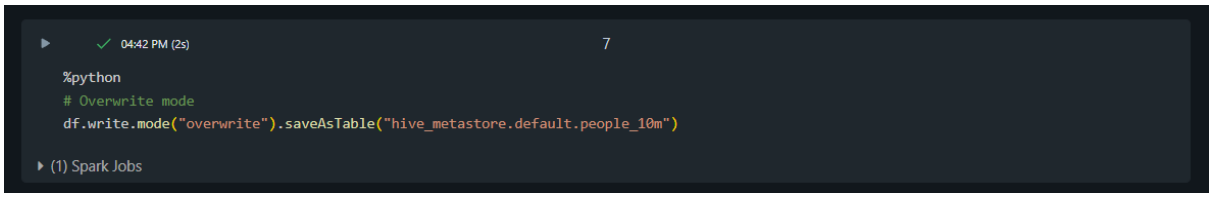
```
%python
# Append mode
df.write.mode("append").saveAsTable("hive_metastore.default.people_10m")
```

04:28 PM (1s) 6

(1) Spark Jobs

Append Mode

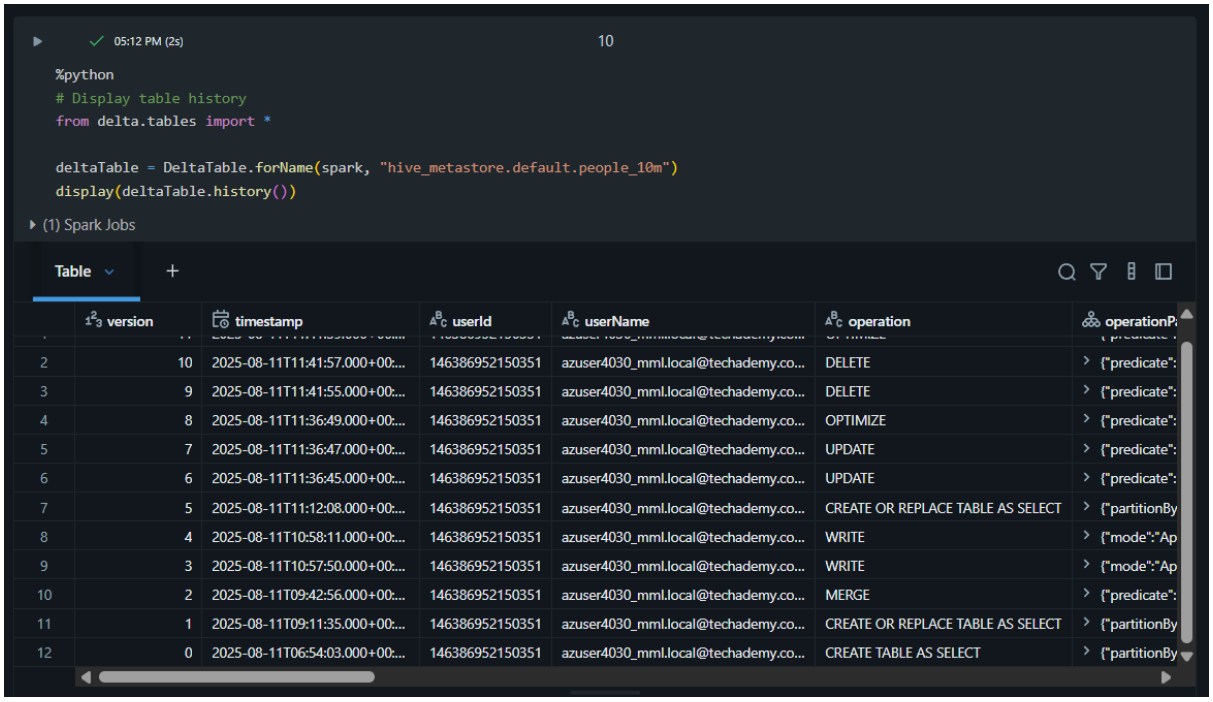
Overwrite Mode



Update a Table

Delete from a Table

Display Table History



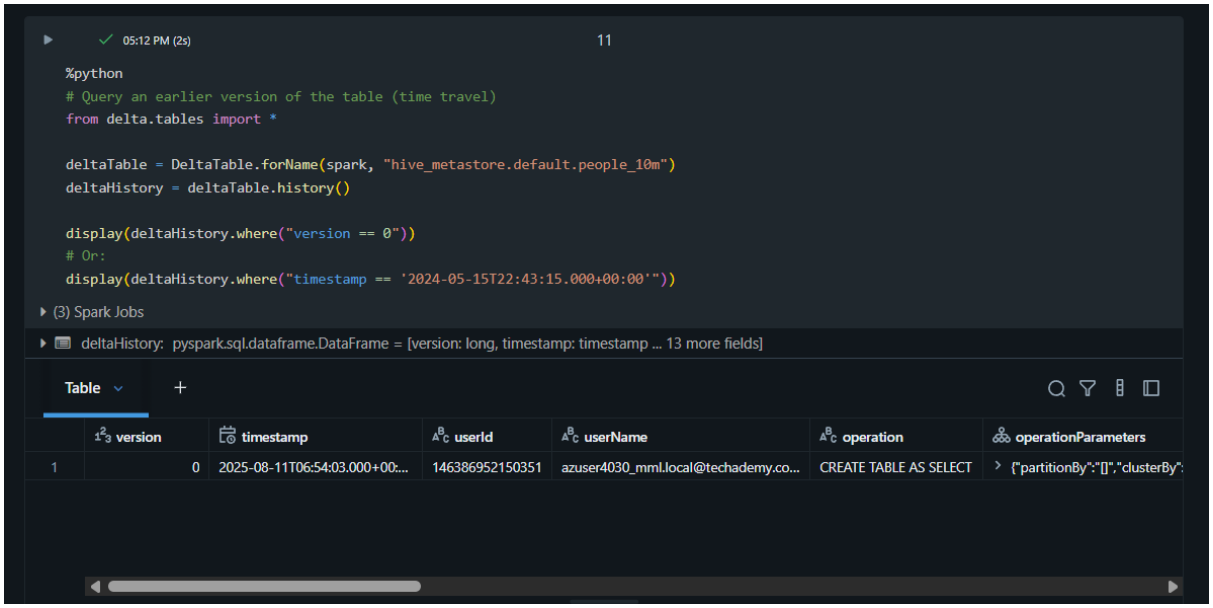
```
%python
# Display table history
from delta.tables import *

deltaTable = DeltaTable.forName(spark, "hive_metastore.default.people_10m")
display(deltaTable.history())
```

(1) Spark Jobs

Table

	version	timestamp	userId	userName	operation	operationP
2	10	2025-08-11T11:41:57.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	DELETE	{"predicate":
3	9	2025-08-11T11:41:55.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	DELETE	{"predicate":
4	8	2025-08-11T11:36:49.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	OPTIMIZE	{"predicate":
5	7	2025-08-11T11:36:47.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	UPDATE	{"predicate":
6	6	2025-08-11T11:36:45.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	UPDATE	{"predicate":
7	5	2025-08-11T11:12:08.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	CREATE OR REPLACE TABLE AS SELECT	{"partitionBy
8	4	2025-08-11T10:58:11.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	WRITE	{"mode":"Ap
9	3	2025-08-11T10:57:50.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	WRITE	{"mode":"Ap
10	2	2025-08-11T09:42:56.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	MERGE	{"predicate":
11	1	2025-08-11T09:11:35.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	CREATE OR REPLACE TABLE AS SELECT	{"partitionBy
12	0	2025-08-11T06:54:03.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	CREATE TABLE AS SELECT	{"partitionBy

Query an earlier version of the table (time travel)



```
%python
# Query an earlier version of the table (time travel)
from delta.tables import *

deltaTable = DeltaTable.forName(spark, "hive_metastore.default.people_10m")
deltaHistory = deltaTable.history()

display(deltaHistory.where("version == 0"))
# Or:
display(deltaHistory.where("timestamp == '2024-05-15T22:43:15.000+00:00'"))
```

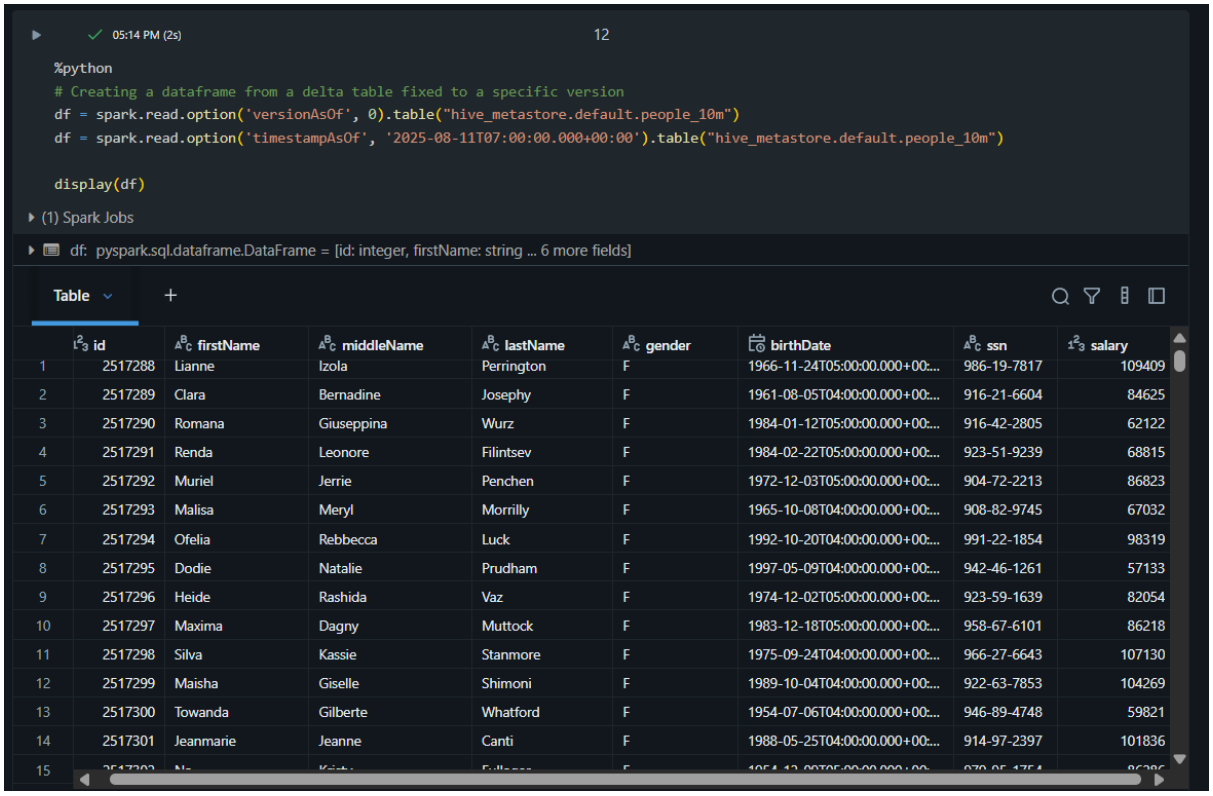
(3) Spark Jobs

deltaHistory: pyspark.sql.dataframe.DataFrame = [version: long, timestamp: timestamp ... 13 more fields]

Table

	version	timestamp	userId	userName	operation	operationParameters
1	0	2025-08-11T06:54:03.000+00:...	146386952150351	azuser4030_mml.local@techademy.co...	CREATE TABLE AS SELECT	{"partitionBy":"[]","clusterBy":

Creating a dataframe from a delta table fixed to a specific version



05:14 PM (2s) 12

```
%python
# Creating a dataframe from a delta table fixed to a specific version
df = spark.read.option('versionAsOf', 0).table("hive_metastore.default.people_10m")
df = spark.read.option('timestampAsOf', '2025-08-11T07:00:00.000+00:00').table("hive_metastore.default.people_10m")

display(df)
```

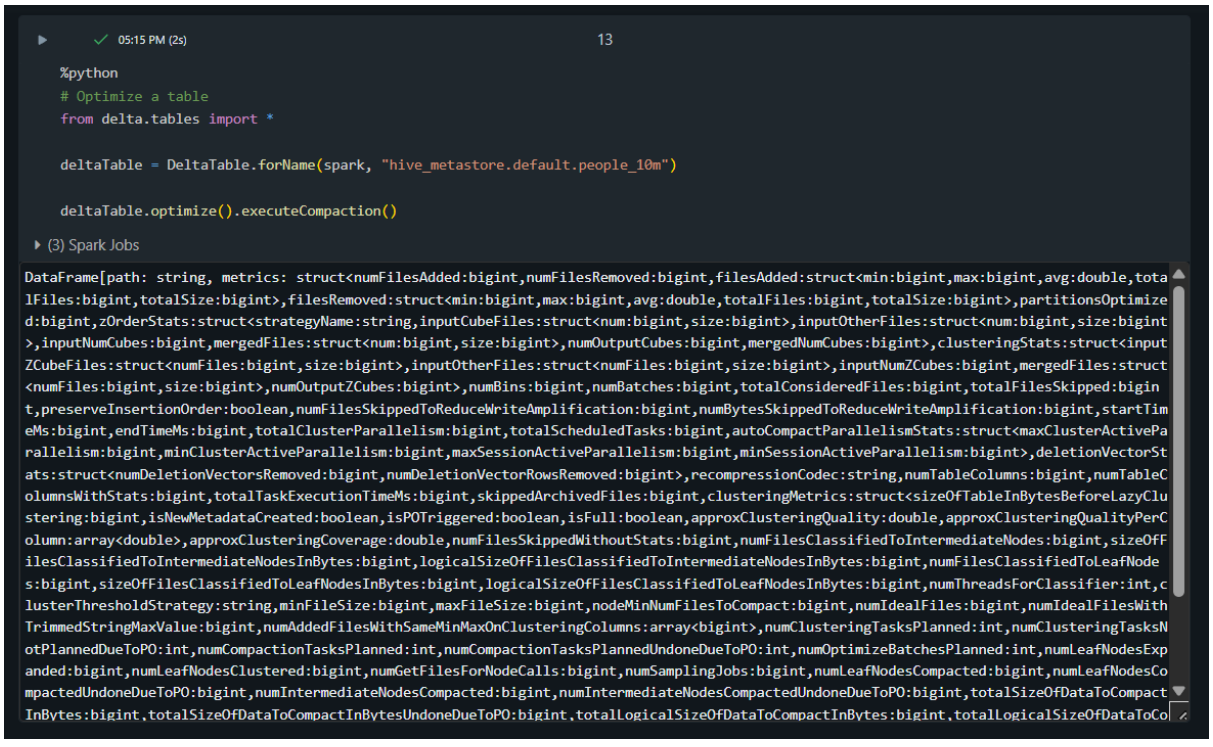
(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [id: integer, firstName: string ... 6 more fields]

Table +

	id	firstName	middleName	lastName	gender	birthDate	ssn	salary
1	2517288	Lianne	Izola	Perrington	F	1966-11-24T05:00:00.000+00:...	986-19-7817	109409
2	2517289	Clara	Bernadine	Josephy	F	1961-08-05T04:00:00.000+00:...	916-21-6604	84625
3	2517290	Romana	Giuseppina	Wurz	F	1984-01-12T05:00:00.000+00:...	916-42-2805	62122
4	2517291	Renda	Leonore	Filintsev	F	1984-02-22T05:00:00.000+00:...	923-51-9239	68815
5	2517292	Muriel	Jerrie	Penchen	F	1972-12-03T05:00:00.000+00:...	904-72-2213	86823
6	2517293	Malisa	Meryl	Morrilly	F	1965-10-08T04:00:00.000+00:...	908-82-9745	67032
7	2517294	Ofelia	Rebbecca	Luck	F	1992-10-20T04:00:00.000+00:...	991-22-1854	98319
8	2517295	Dodie	Natalie	Prudham	F	1997-05-09T04:00:00.000+00:...	942-46-1261	57133
9	2517296	Heide	Rashida	Vaz	F	1974-12-02T05:00:00.000+00:...	923-59-1639	82054
10	2517297	Maxima	Dagny	Muttock	F	1983-12-18T05:00:00.000+00:...	958-67-6101	86218
11	2517298	Silva	Kassie	Stanmore	F	1975-09-24T04:00:00.000+00:...	966-27-6643	107130
12	2517299	Maisha	Giselle	Shimoni	F	1989-10-04T04:00:00.000+00:...	922-63-7853	104269
13	2517300	Towanda	Gilberte	Whatford	F	1954-07-06T04:00:00.000+00:...	946-89-4748	59821
14	2517301	Jeanmarie	Jeanne	Canti	F	1988-05-25T04:00:00.000+00:...	914-97-2397	101836
15								

Optimizing a Table



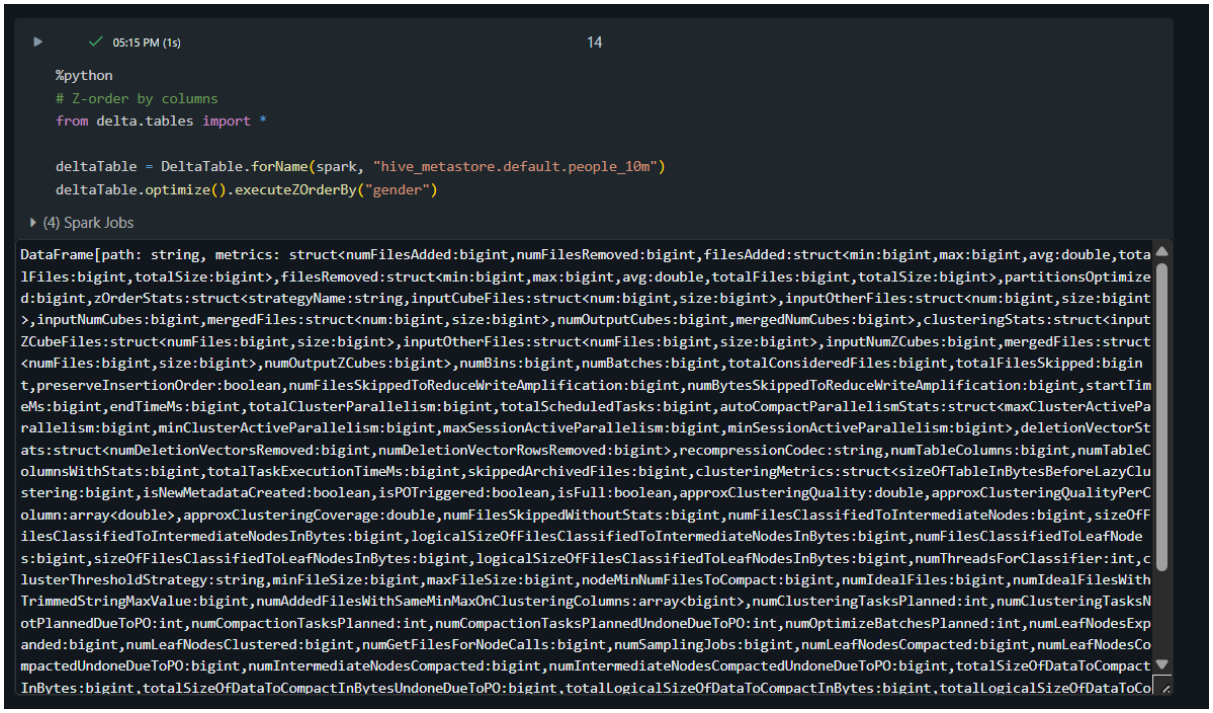
05:15 PM (2s) 13

```
%python
# Optimize a table
from delta.tables import *

deltaTable = DeltaTable.forName(spark, "hive_metastore.default.people_10m")

deltaTable.optimize().executeCompaction()
```

(3) Spark Jobs

```
DataFrame[path: string, metrics: struct<numFilesAdded:bigint,numFilesRemoved:bigint,filesAdded:struct<min:bigint,max:bigint,avg:double,tota
lFiles:bigint,totalSize:bigint>,filesRemoved:struct<min:bigint,max:bigint,avg:double,totalFiles:bigint,totalSize:bigint>,partitionsOptimize
d:bigint,zOrderStats:struct<strategyName:string,inputCubeFiles:struct<num:bigint,size:bigint>,inputOtherFiles:struct<num:bigint,size:bigint
>,inputNumCubes:bigint,mergedFiles:struct<num:bigint,size:bigint>,numOutputCubes:bigint,mergedNumCubes:bigint>,clusteringStats:struct<input
ZCubeFiles:struct<numFiles:bigint,size:bigint>,inputOtherFiles:struct<numFiles:bigint,size:bigint>,inputNumZCubes:bigint,mergedFiles:struct
<numFiles:bigint,size:bigint>,numOutputZCubes:bigint>,numBins:bigint,numBatches:bigint,totalConsideredFiles:bigint,totalFilesSkipped:bigin
t,preserveInsertionOrder:boolean,numFilesSkippedToReduceWriteAmplification:bigint,numBytesSkippedToReduceWriteAmplification:bigint,startTim
eMs:bigint,endTimeMs:bigint,totalClusterParallelism:bigint,totalScheduledTasks:bigint,autoCompactParallelismStats:struct<maxClusterActivePa
rallelism:bigint,minClusterActiveParallelism:bigint,maxSessionActiveParallelism:bigint,minSessionActiveParallelism:bigint>,deletionVectorSt
ats:struct<numDeletionVectorsRemoved:bigint,numDeletionVectorRowsRemoved:bigint>,recompressionCodec:string,numTableColumns:bigint,numTableC
olumnsWithStats:bigint,totalTaskExecutionTimeMs:bigint,skippedArchivedFiles:bigint,clusteringMetrics:struct<sizeOfTableInBytesBeforeLazyClu
stering:bigint,isNewMetadataCreated:boolean,isPOTriggered:boolean,isFull:boolean,approxClusteringQuality:double,approxClusteringQualityPerC
olumn:array<double>,approxClusteringCoverage:double,numFilesSkippedWithoutStats:bigint,numFilesClassifiedToIntermediateNodes:bigint,sizeOfF
ilesClassifiedToIntermediateNodesInBytes:bigint,logicalSizeOfFilesClassifiedToIntermediateNodesInBytes:bigint,numFilesClassifiedToLeafNode
s:bigint,sizeOfFilesClassifiedToLeafNodesInBytes:bigint,logicalSizeOfFilesClassifiedToLeafNodesInBytes:bigint,numThreadsForClassifier:int,c
lusterThresholdStrategy:string,minFileSize:bigint,maxFileSize:bigint,nodeMinNumFilesToCompact:bigint,numIdealFiles:bigint,numIdealFilesWith
TrimmedStringMaxValue:bigint,numAddedFilesWithSameMinMaxOnClusteringColumns:array<bigint>,numClusteringTasksPlanned:int,numClusteringTasksN
otPlannedDueToPO:int,numCompactionTasksPlanned:int,numCompactionTasksPlannedUndoneDueToPO:int,numOptimizeBatchesPlanned:int,numLeafNodesExp
anded:bigint,numLeafNodesClustered:bigint,numGetFilesForNodeCalls:bigint,numSamplingJobs:bigint,numLeafNodesCompacted:bigint,numLeafNodesCo
mpactedUndoneDueToPO:bigint,numIntermediateNodesCompacted:bigint,numIntermediateNodesCompactedUndoneDueToPO:bigint,totalSizeOfDataToCompact
InBytes:bigint,totalSizeOfDataToCompactInBytesUndoneDueToPO:bigint,totalLogicalSizeOfDataToCompactInBytes:bigint,totalLogicalSizeOfDataToCo
```

Z-order by columns



Clean up snapshots with VACUUM