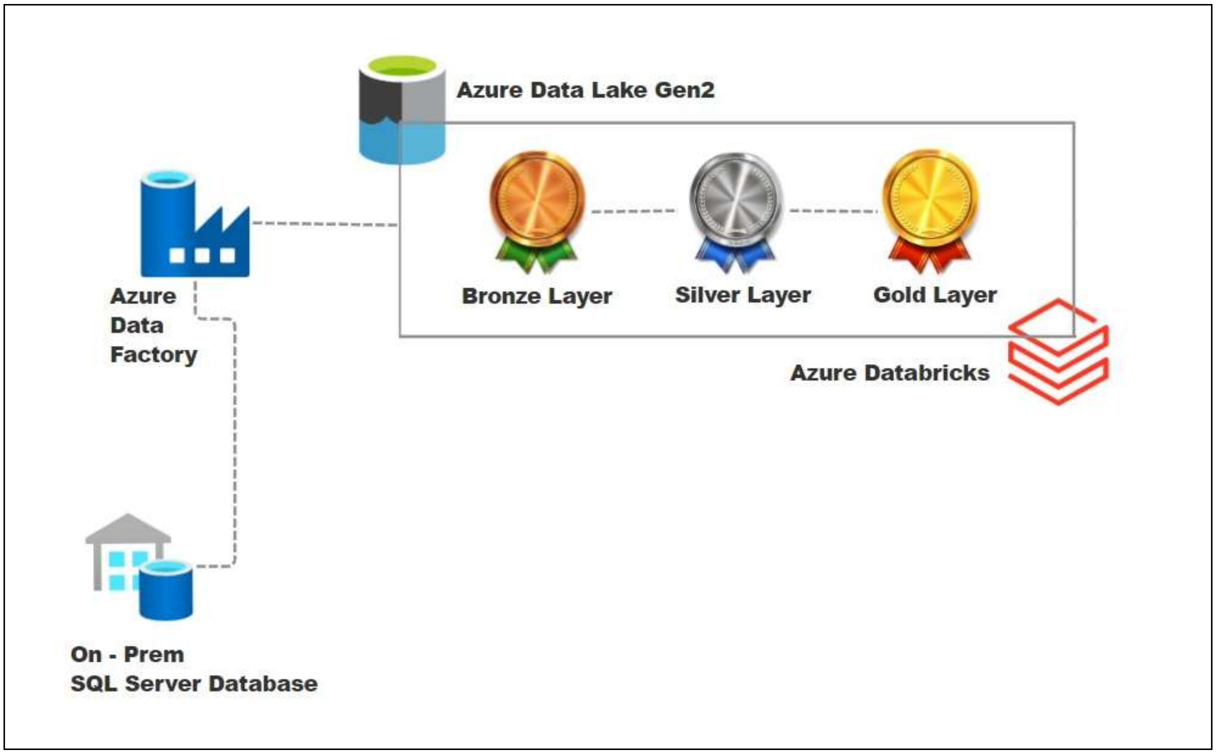
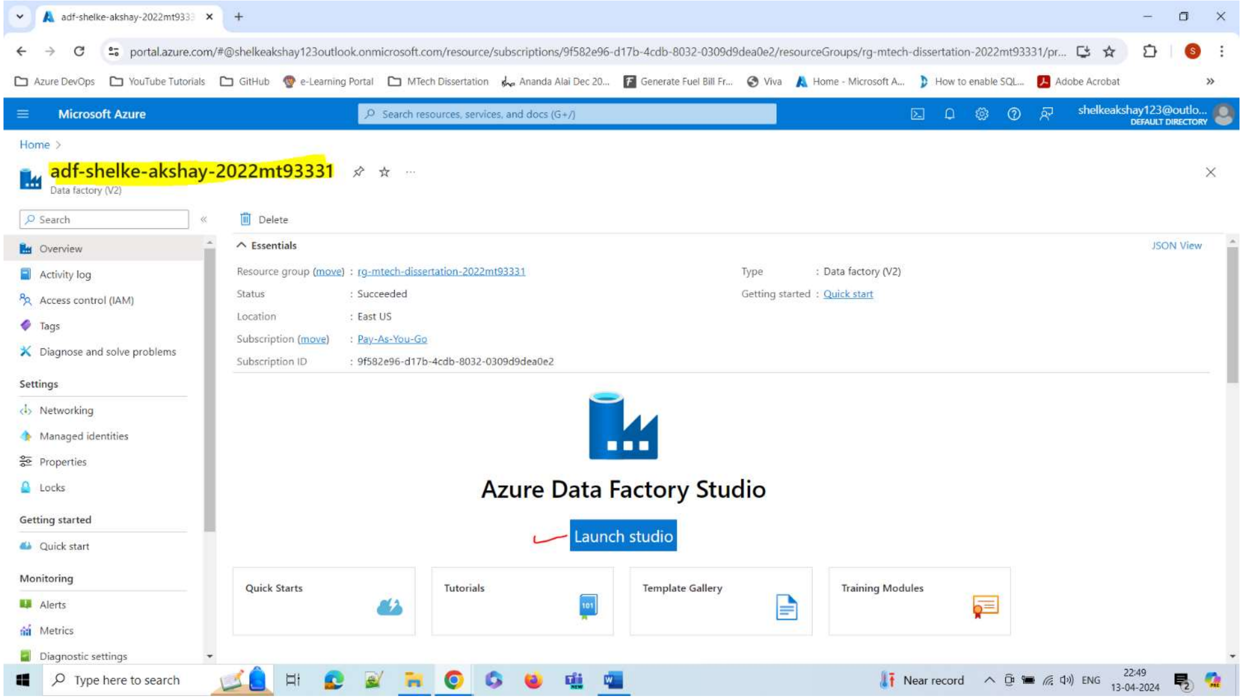
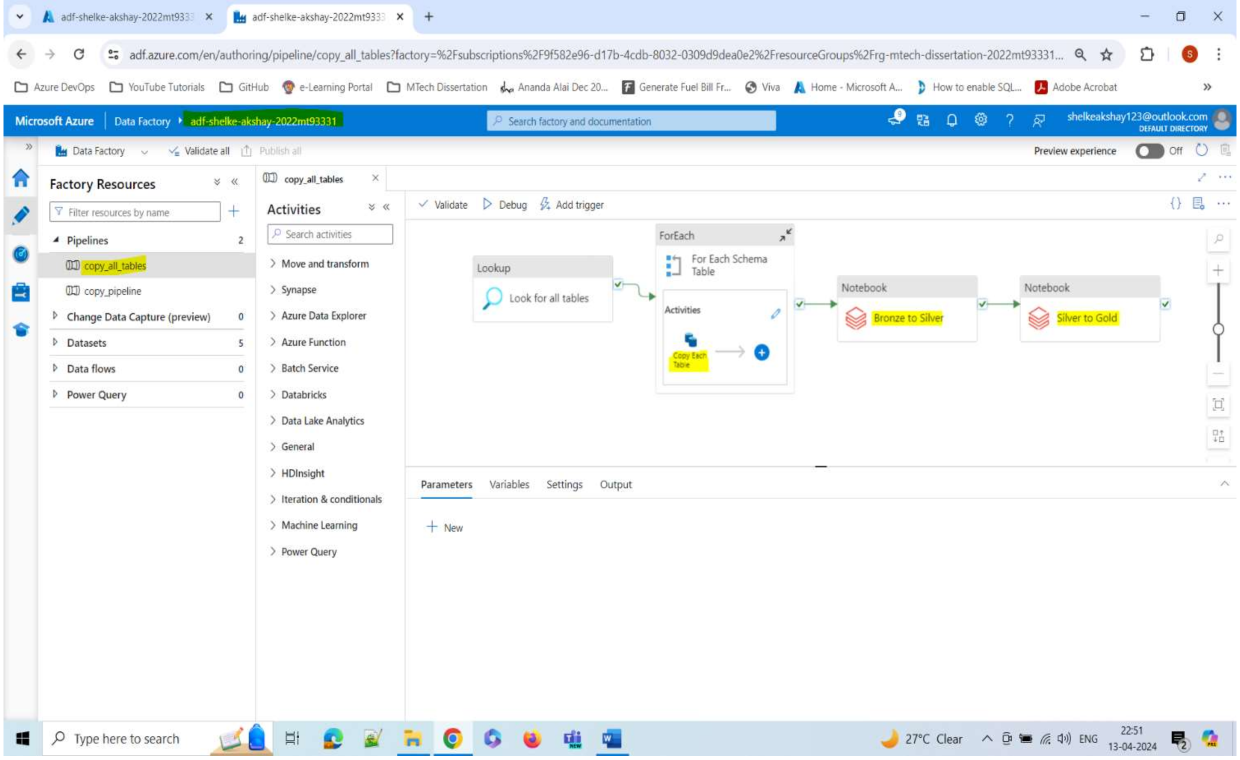
Data Transformation

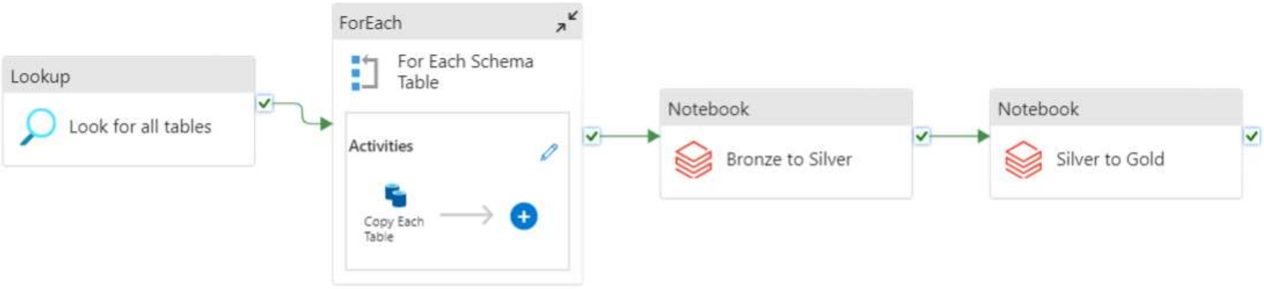


1) Azure Data Factory(ADF) Pipeline



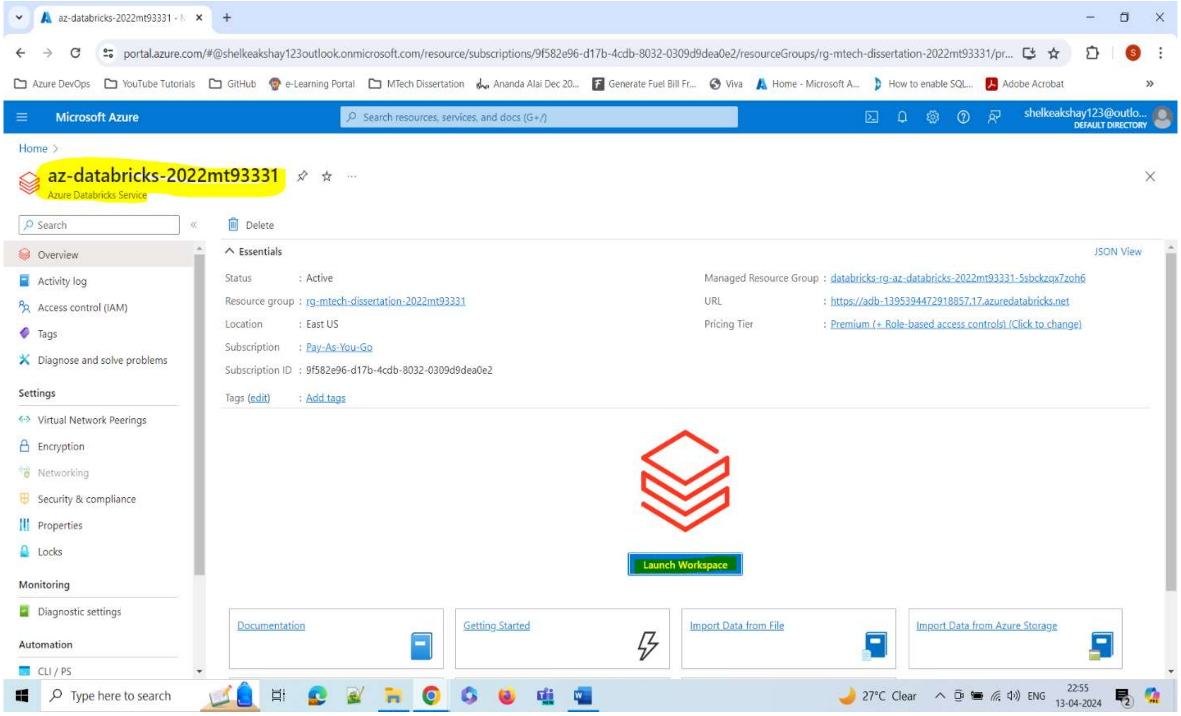
Resource group (move) : rg-mtech-dissertation-2022mt93331
Status : Succeeded
Location : East US
Subscription (move) : Pay-As-You-Go
Subscription ID : 9f582e96-d17b-4cdb-8032-0309d9dea0e2
Type : Data factory (V2)
Getting started : Quick start





a. Look for all tables.
b. Copy Each table into BRONZE container.
c. Data Transformation – BRONZE to SILVER
d. Data Transformation – SILVER to GOLD

2) Azure Databricks(ADB) – Code Notebooks for Data Transformation

Azure Databricks Instance


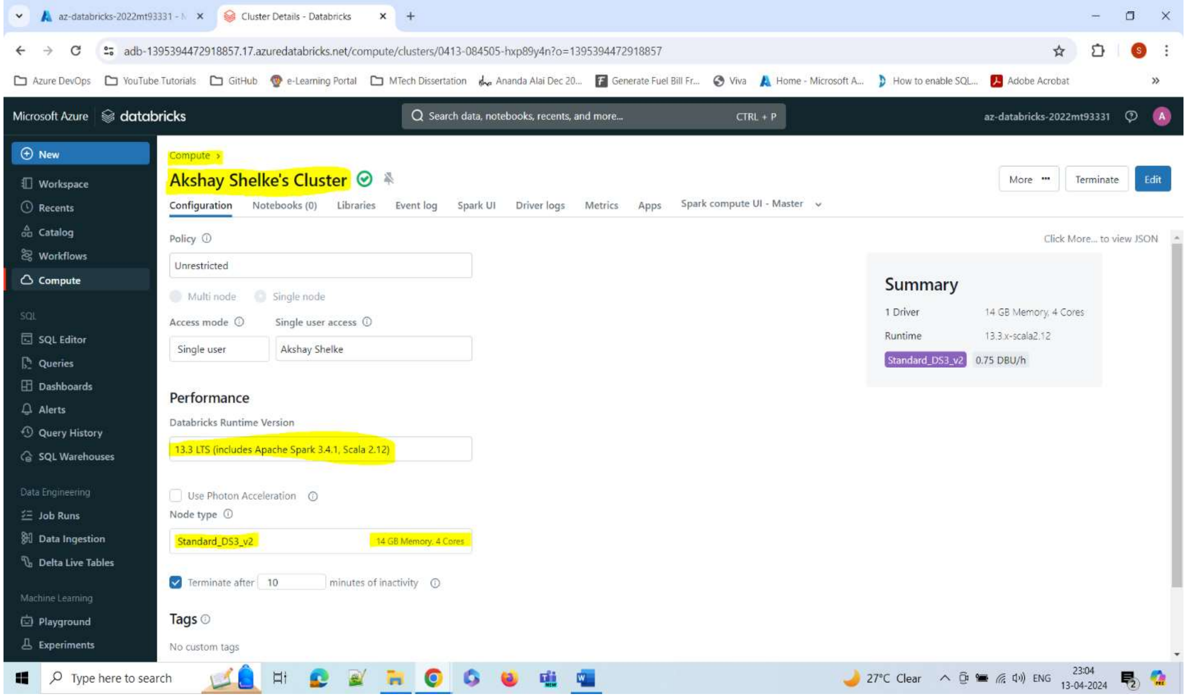
portal.azure.com/#@shelkeakshay123outlook.onmicrosoft.com/resource/subscriptions/9f582e96-d17b-4cdb-8032-0309d9dea0e2/resourceGroups/rg-mtech-dissertation-2022mt93331/pr...

az-databricks-2022mt93331
Azure Databricks Service

Essentials

Status	: Active
Resource group	: rg-mtech-dissertation-2022mt93331
Location	: East US
Subscription	: Pay-As-You-Go
Subscription ID	: 9f582e96-d17b-4cdb-8032-0309d9dea0e2
Tags (edit)	: Add tags
Managed Resource Group	: databricks-rg-az-databricks-2022mt93331-5sbckzqx7zoh6
URL	: https://adb-1395394472918857.17.azuredatabricks.net
Pricing Tier	: Premium (+ Role-based access controls) (Click to change)

Azure Databricks Compute Cluster



adb-1395394472918857.17.azuredatabricks.net/compute/clusters/0413-084505-hxp89y4n?o=1395394472918857

Compute ›

Akshay Shelke's Cluster

Policy: Unrestricted

○ Multi node ○ Single node

Access mode: Single user

Single user access: Akshay Shelke

Performance

Databricks Runtime Version: 13.3 LTS (includes Apache Spark 3.4.1, Scala 2.12)

□ Use Photon Acceleration

Node type: Standard_DS3_v2 14 GB Memory, 4 Cores

☑ Terminate after 10 minutes of inactivity

Tags

No custom tags

Summary

1 Driver	14 GB Memory, 4 Cores
Runtime	13.3.x-scala2.12
Standard_DS3_v2	0.75 DBU/h

Python Code To create Mount point for Azure Blob Storage Container – For Bronze Container

```
configs = {
  "fs.azure.account.auth.type": "CustomAccessToken",
  "fs.azure.account.custom.token.provider.class":
spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClassName")
}

try:
  dbutils.fs.mount(
    source = "abfss://bronze@storageaccmtechdemo.dfs.core.windows.net/",
    mount_point = "/mnt/bronze",
    extra_configs = configs)
  print("Mount Point created successfully")
except:
  print("Mount Point already exists")
```

Python Code To create Mount point for Azure Blob Storage Container – For Silver Container

```python
configs = {
  "fs.azure.account.auth.type": "CustomAccessToken",
  "fs.azure.account.custom.token.provider.class":
spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClassName")
}

try:
  dbutils.fs.mount(
    source = "abfss://silver@storageaccmtechdemo.dfs.core.windows.net/",
    mount_point = "/mnt/silver",
    extra_configs = configs)
  print("Mount Point created successfully")
except:
  print("Mount Point already exists")
```

Python Code To create Mount point for Azure Blob Storage Container – For Gold Container

```python
configs = {
  "fs.azure.account.auth.type": "CustomAccessToken",
  "fs.azure.account.custom.token.provider.class":
spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClassName")
}

try:
  dbutils.fs.mount(
    source = "abfss://gold@storageaccmtechdemo.dfs.core.windows.net/",
    mount_point = "/mnt/gold",
    extra_configs = configs)
  print("Mount Point created successfully")
except:
  print("Mount Point already exists")
```

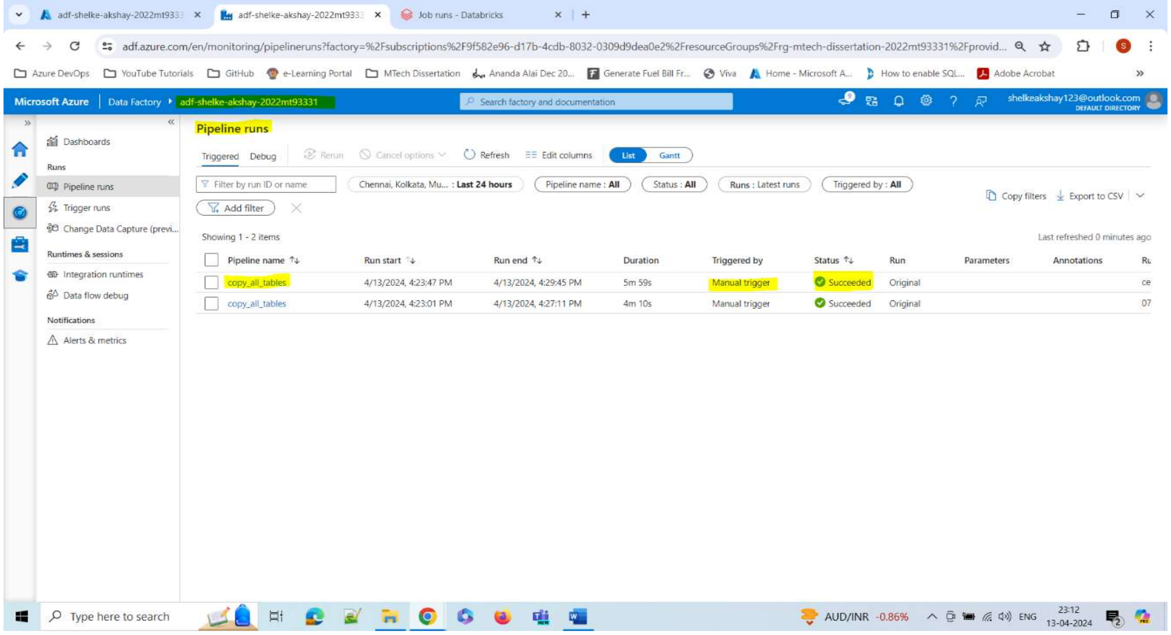
Python Code for Data Transformation – BRONZE to SILVER

```python
from pyspark.sql.functions import from_utc_timestamp, date_format
from pyspark.sql.types import TimestampType


dbutils.fs.ls("mnt/bronze/SalesLT/")


table_name = []

for i in dbutils.fs.ls("mnt/bronze/SalesLT/"):
    table_name.append(i.name.split('/')[0])



for i in table_name:
    path = "/mnt/bronze/SalesLT/" + i + "/" + i + ".parquet"
    df = spark.read.format("parquet").load(path)
    column = df.columns

    for col in column:
        if "Date" in col or "date" in col:
            df = df.withColumn(col,
date_format(from_utc_timestamp(df[col].cast(TimestampType()), "UTC"), "yyyy-MM-dd"))

    output_path = "/mnt/silver/SalesLT/" + i + "/"
    df.write.format("delta").mode("overwrite").save(output_path)

display(df)
```

Python Code for Data Transformation – SILVER to GOLD

```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, regexp_replace


dbutils.fs.ls("mnt/silver/SalesLT/")

table_name = []

for i in dbutils.fs.ls("mnt/silver/SalesLT/"):
    table_name.append(i.name.split('/')[0])


for name in table_name:
    path = "/mnt/silver/SalesLT/" + name
    print(path)
    df = spark.read.format("delta").load(path)

    # Get the list of column names
    column_names = df.columns

    for old_col_name in column_names:
        # Convert column name from ColumnName to Column_Name format
        new_col_name = "".join(["_" + char if char.isupper() and not old_col_name[i-1].isupper() else char for i, char in enumerate(old_col_name)]).lstrip("_")

        # Change the column name using withColumnRenamed and regexp_replace
        df = df.withColumnRenamed(old_col_name, new_col_name)

    output_path = "/mnt/gold/SalesLT/" + name + "/"
    df.write.format("delta").mode("overwrite").save(output_path)

display(df)
```

3) Azure Data Factory pipeline RUNS



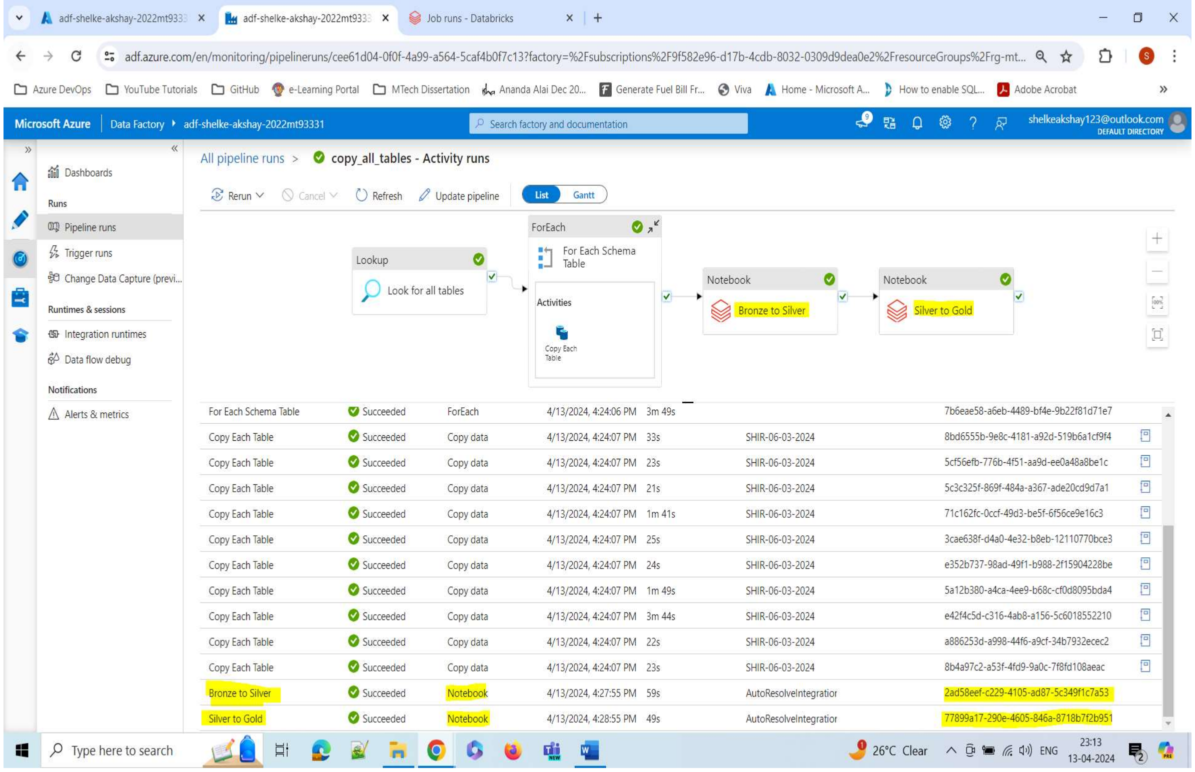
Pipeline runs

Chennai, Kolkata, Mu... : **Last 24 hours** | Pipeline name : **All** | Status : **All** | Runs : Latest runs | Triggered by : **All**

Showing 1 - 2 items

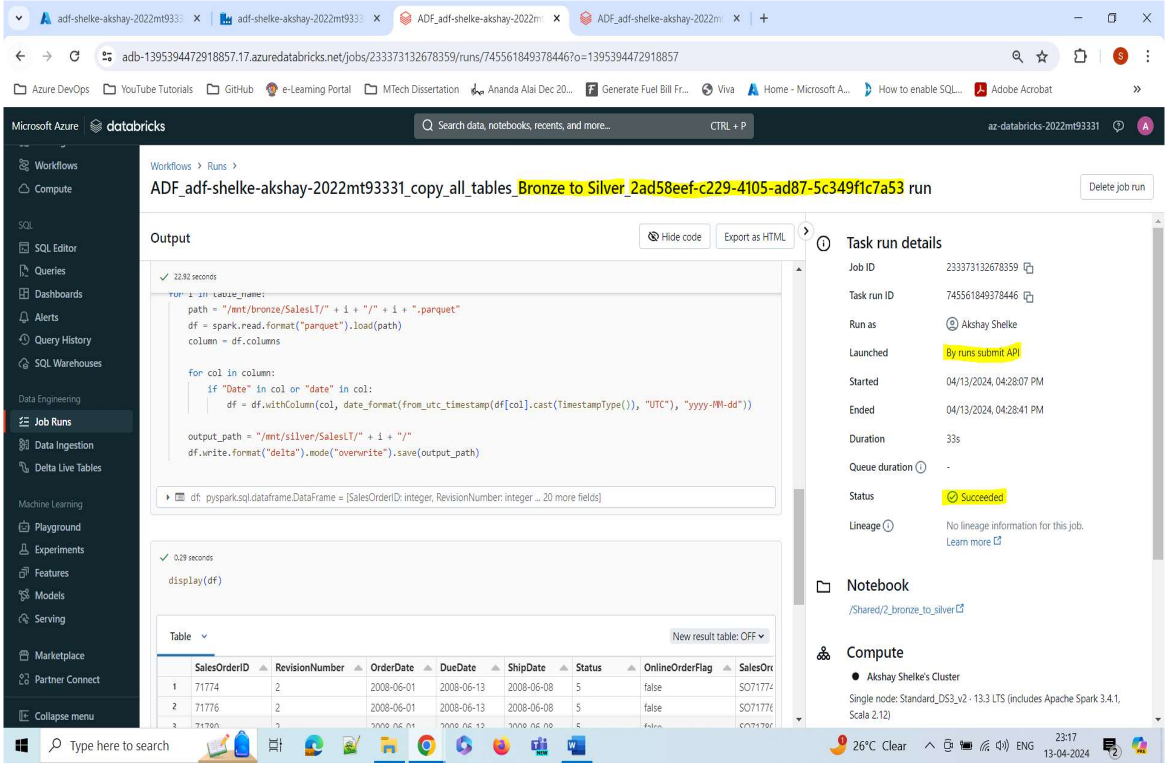
Last refreshed 0 minutes ago

Pipeline name	Run start	Run end	Duration	Triggered by	Status	Run	Parameters	Annotations	Ru
copy_all_tables	4/13/2024, 4:23:47 PM	4/13/2024, 4:29:45 PM	5m 59s	Manual trigger	✓ Succeeded	Original			ce
copy_all_tables	4/13/2024, 4:23:01 PM	4/13/2024, 4:27:11 PM	4m 10s	Manual trigger	✓ Succeeded	Original			07



All pipeline runs > ✓ copy_all_tables - Activity runs

For Each Schema Table	✓ Succeeded	ForEach	4/13/2024, 4:24:06 PM	3m 49s		7b6eae58-a6eb-4489-bf4e-9b22f81d71e7
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	33s	SHIR-06-03-2024	8bd6555b-9e8c-4181-a92d-519b6a1cf9f4
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	23s	SHIR-06-03-2024	5cf56efb-776b-4f51-aa9d-ee0a48a8be1c
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	21s	SHIR-06-03-2024	5c3c325f-869f-484a-a367-ade20cd9d7a1
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	1m 41s	SHIR-06-03-2024	71c162fc-0ccf-49d3-be5f-6f56ce9e16c3
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	25s	SHIR-06-03-2024	3cae638f-d4a0-4e32-b8eb-12110770bce3
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	24s	SHIR-06-03-2024	e352b737-98ad-49f1-b988-2f15904228be
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	1m 49s	SHIR-06-03-2024	5a12b380-a4ca-4ee9-b68c-cf0d8095bda4
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	3m 44s	SHIR-06-03-2024	e42f4c5d-c316-4ab8-a156-5c6018552210
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	22s	SHIR-06-03-2024	a886253d-a998-44f6-a9cf-34b7932ecec2
Copy Each Table	✓ Succeeded	Copy data	4/13/2024, 4:24:07 PM	23s	SHIR-06-03-2024	8b4a97c2-a53f-4fd9-9a0c-7f8fd108aeac
Bronze to Silver	✓ Succeeded	Notebook	4/13/2024, 4:27:55 PM	59s	AutoResolveIntegratior	2ad58eef-c229-4105-ad87-5c349f1c7a53
Silver to Gold	✓ Succeeded	Notebook	4/13/2024, 4:28:55 PM	49s	AutoResolveIntegratior	77899a17-290e-4605-846a-8718b7f2b951



ADF_adf-shelke-akshay-2022mt93331_copy_all_tables_Bronze to Silver_2ad58eef-c229-4105-ad87-5c349f1c7a53 run

Output

✓ 22.92 seconds

```
for i in table_name:
    path = "/mnt/bronze/SalesLT/" + i + "/" + i + ".parquet"
    df = spark.read.format("parquet").load(path)
    column = df.columns

    for col in column:
        if "Date" in col or "date" in col:
            df = df.withColumn(col, date_format(from_utc_timestamp(df[col].cast(TimestampType()), "UTC"), "yyyy-MM-dd"))

    output_path = "/mnt/silver/SalesLT/" + i + "/"
    df.write.format("delta").mode("overwrite").save(output_path)
```

df: pyspark.sql.dataframe.DataFrame = [SalesOrderID: integer, RevisionNumber: integer ... 20 more fields]

✓ 0.29 seconds

```
display(df)
```

	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrc
1	71774	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71774
2	71776	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71776
3	71780	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO7178[illegible]

Task run details

Job ID	233373132678359
Task run ID	745561849378446
Run as	Akshay Shelke
Launched	By runs submit API
Started	04/13/2024, 04:28:07 PM
Ended	04/13/2024, 04:28:41 PM
Duration	33s
Queue duration	-
Status	Succeeded
Lineage	No lineage information for this job. Learn more

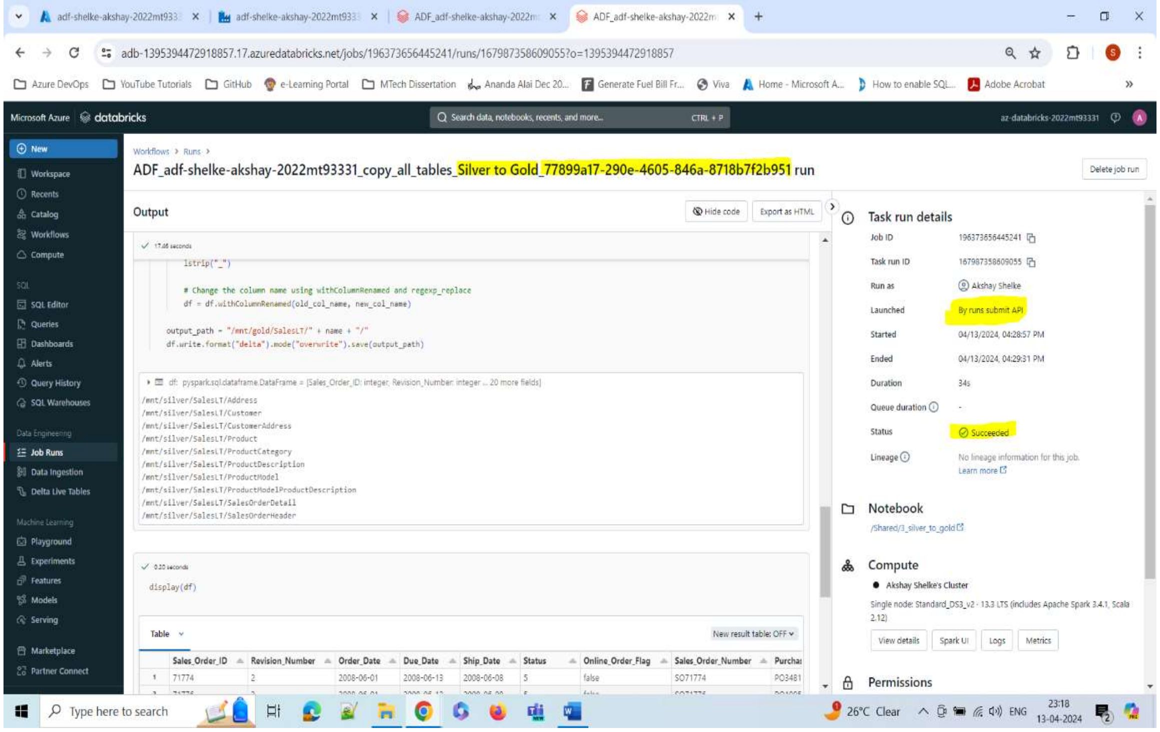
Notebook

/Shared/2_bronze_to_silver

Compute

- Akshay Shelke's Cluster

Single node: Standard_DS3_v2 · 13.3 LTS (includes Apache Spark 3.4.1, Scala 2.12)



adb-1395394472918857.17.azuredatabricks.net/jobs/196373656445241/runs/167987358609055?o=1395394472918857

Workflows > Runs >

ADF_adf-shelke-akshay-2022mt93331_copy_all_tables_Silver to Gold_77899a17-290e-4605-846a-8718b7f2b951 run

Output

✓ 17.46 seconds

```
        lstrip("_")

        # Change the column name using withColumnRenamed and regexp_replace
        df = df.withColumnRenamed(old_col_name, new_col_name)

    output_path = "/mnt/gold/SalesLT/" + name + "/"
    df.write.format("delta").mode("overwrite").save(output_path)
```

```
df: pyspark.sql.dataframe.DataFrame = [Sales_Order_ID: integer, Revision_Number: integer ... 20 more fields]
/mnt/silver/SalesLT/Address
/mnt/silver/SalesLT/Customer
/mnt/silver/SalesLT/CustomerAddress
/mnt/silver/SalesLT/Product
/mnt/silver/SalesLT/ProductCategory
/mnt/silver/SalesLT/ProductDescription
/mnt/silver/SalesLT/ProductModel
/mnt/silver/SalesLT/ProductModelProductDescription
/mnt/silver/SalesLT/SalesOrderDetail
/mnt/silver/SalesLT/SalesOrderHeader
```

✓ 0.20 seconds

```
display(df)
```

Table

New result table: OFF

	Sales_Order_ID	Revision_Number	Order_Date	Due_Date	Ship_Date	Status	Online_Order_Flag	Sales_Order_Number	Purchas
1	71774	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71774	PO3481

Task run details

Job ID	196373656445241
Task run ID	167987358609055
Run as	Akshay Shelke
Launched	By runs submit API
Started	04/13/2024, 04:28:57 PM
Ended	04/13/2024, 04:29:31 PM
Duration	34s
Queue duration	-
Status	Succeeded
Lineage	No lineage information for this job. Learn more

Notebook

/Shared/3_silver_to_gold

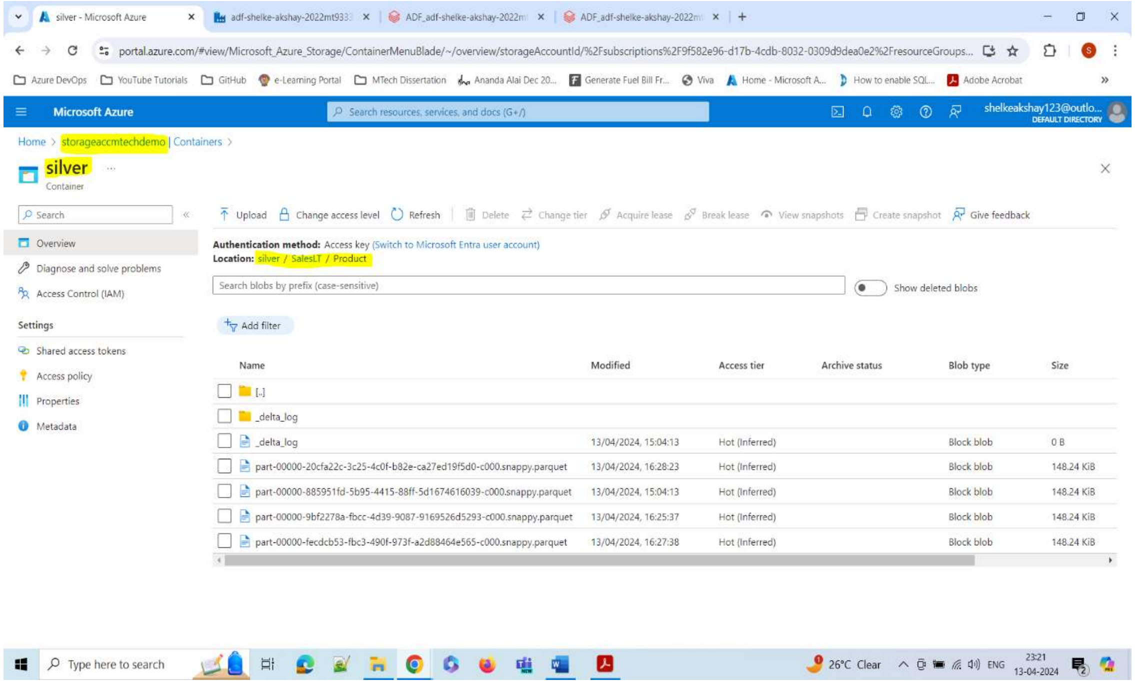
Compute

● Akshay Shelke's Cluster

Single node: Standard_DS3_v2 · 13.3 LTS (includes Apache Spark 3.4.1, Scala 2.12)

Permissions

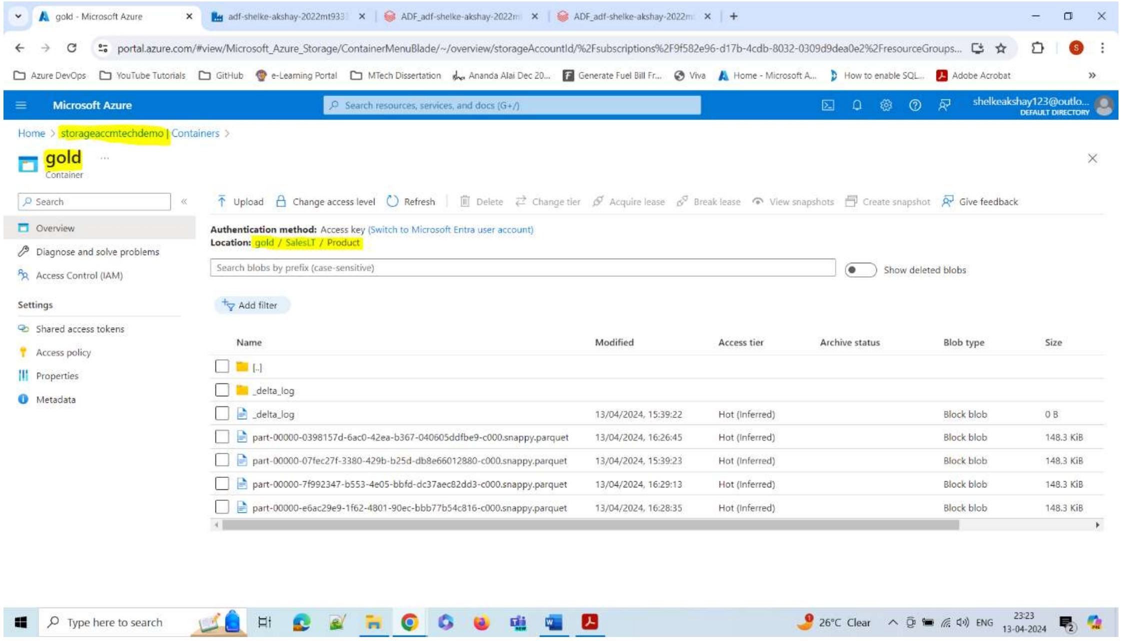
4) Delta tables stored in Silver Container



Authentication method: Access key (Switch to Microsoft Entra user account)
Location: silver / SalesLT / Product

Name	Modified	Access tier	Archive status	Blob type	Size
[..]					
_delta_log					
_delta_log	13/04/2024, 15:04:13	Hot (Inferred)		Block blob	0 B
part-00000-20cfa22c-3c25-4c0f-b82e-ca27ed19f5d0-c000.snappy.parquet	13/04/2024, 16:28:23	Hot (Inferred)		Block blob	148.24 KiB
part-00000-885951fd-5b95-4415-88ff-5d1674616039-c000.snappy.parquet	13/04/2024, 15:04:13	Hot (Inferred)		Block blob	148.24 KiB
part-00000-9bf2278a-fbcc-4d39-9087-9169526d5293-c000.snappy.parquet	13/04/2024, 16:25:37	Hot (Inferred)		Block blob	148.24 KiB
part-00000-fecdcb53-fbc3-490f-973f-a2d88464e565-c000.snappy.parquet	13/04/2024, 16:27:38	Hot (Inferred)		Block blob	148.24 KiB

5) Delta tables stored in Gold Container – Data Transformation Completed



Authentication method: Access key (Switch to Microsoft Entra user account)
Location: gold / SalesLT / Product

Name	Modified	Access tier	Archive status	Blob type	Size
[..]					
_delta_log					
_delta_log	13/04/2024, 15:39:22	Hot (Inferred)		Block blob	0 B
part-00000-0398157d-6ac0-42ea-b367-040605ddfbe9-c000.snappy.parquet	13/04/2024, 16:26:45	Hot (Inferred)		Block blob	148.3 KiB
part-00000-07fec27f-3380-429b-b25d-db8e66012880-c000.snappy.parquet	13/04/2024, 15:39:23	Hot (Inferred)		Block blob	148.3 KiB
part-00000-7f992347-b553-4e05-bbfd-dc37aec82dd3-c000.snappy.parquet	13/04/2024, 16:29:13	Hot (Inferred)		Block blob	148.3 KiB
part-00000-e6ac29e9-1f62-4801-90ec-bbb77b54c816-c000.snappy.parquet	13/04/2024, 16:28:35	Hot (Inferred)		Block blob	148.3 KiB