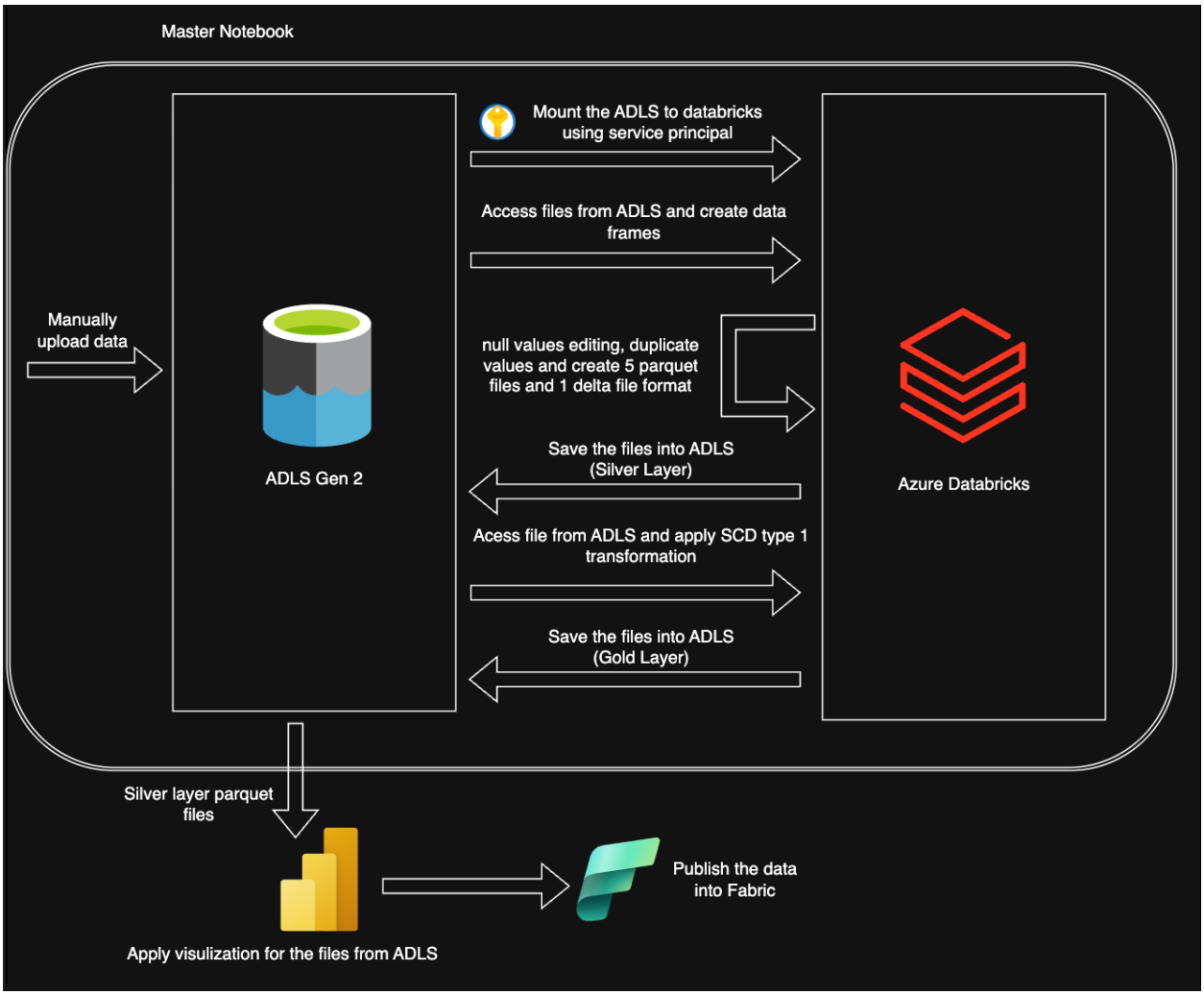
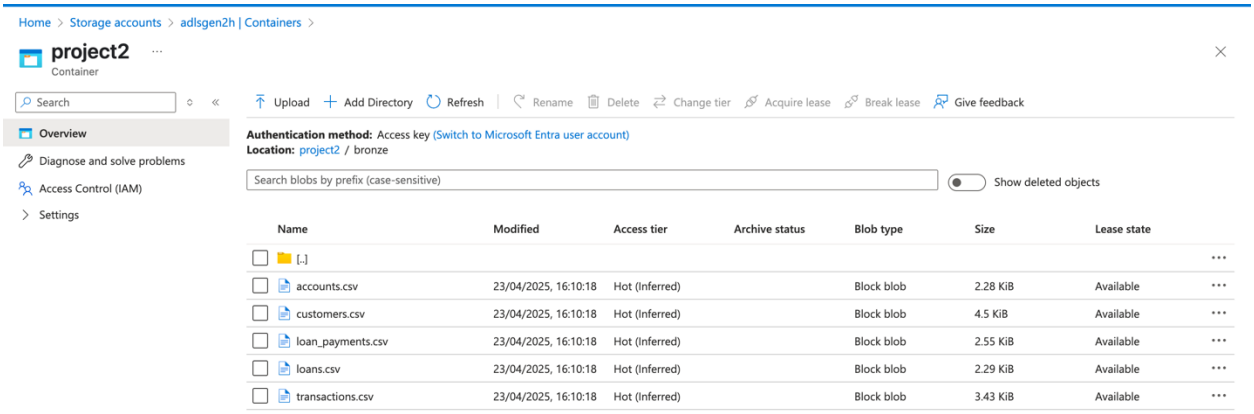
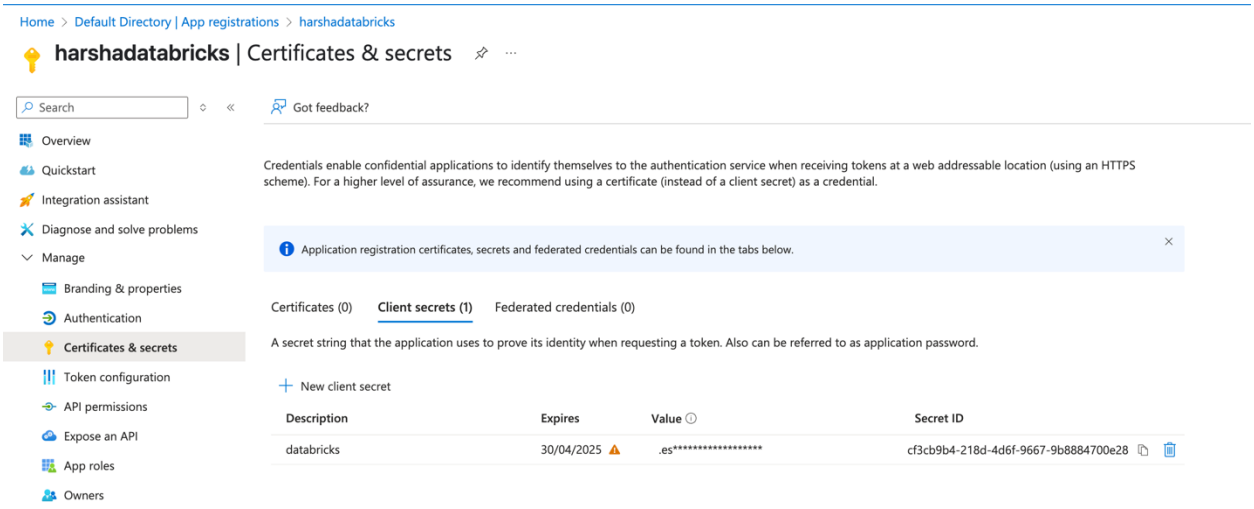
PROJECT 2



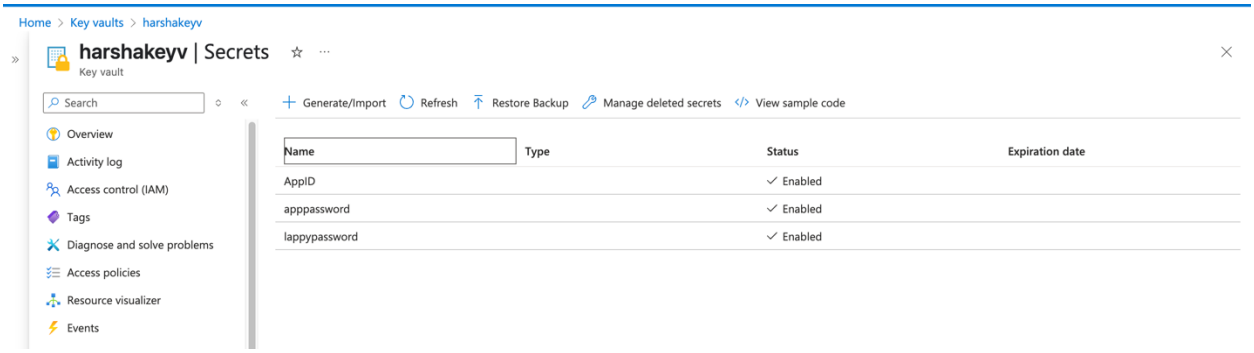
Architecture Diagram



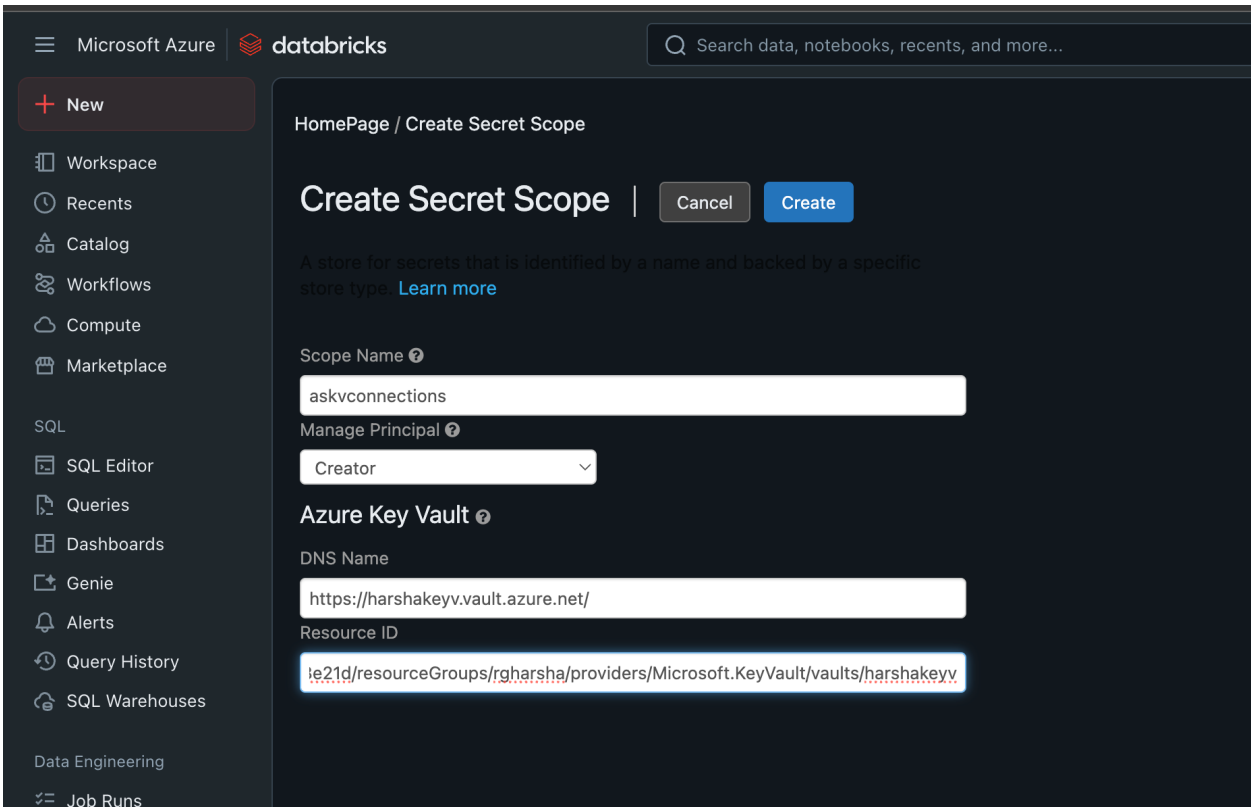
Uploaded the files



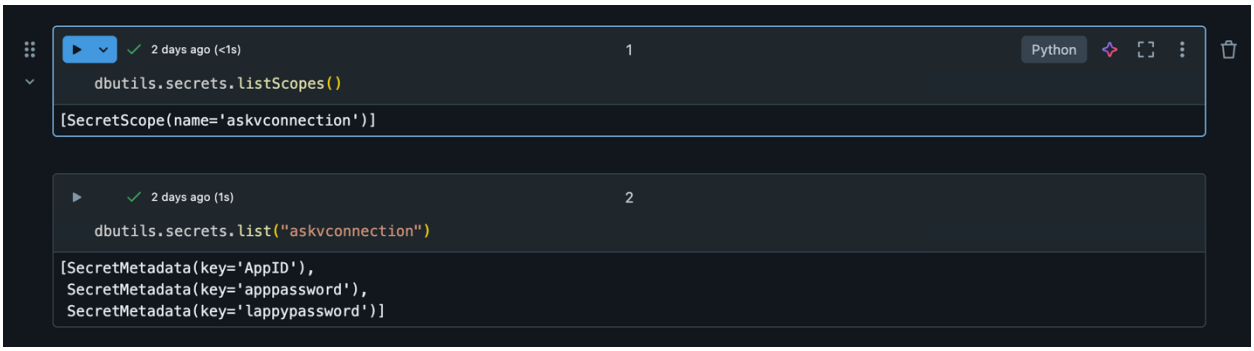
Created service principal for databricks



Created secrets using app id and password in keyvault to access adls



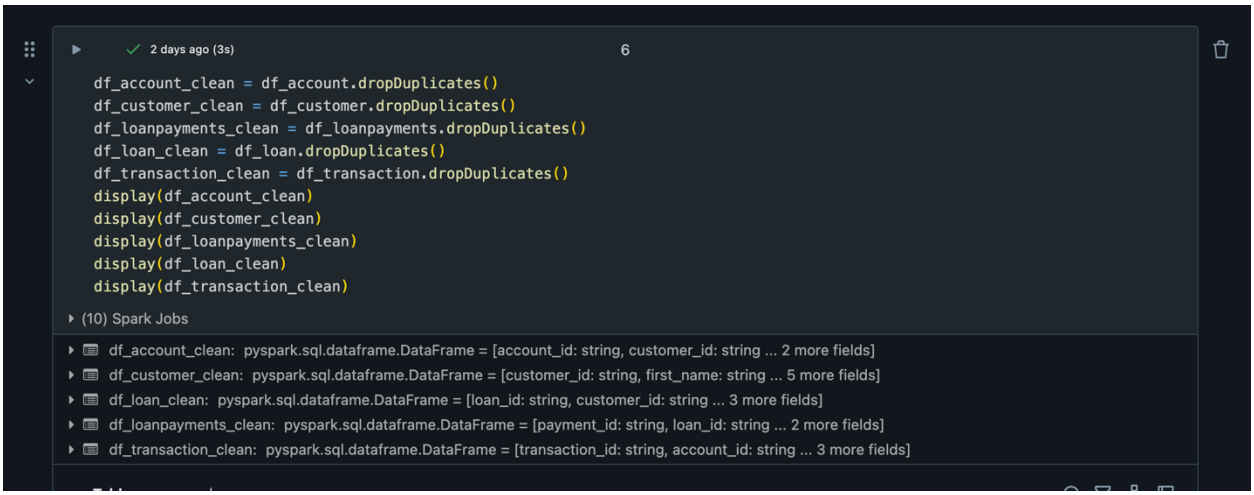
Created scope in databricks



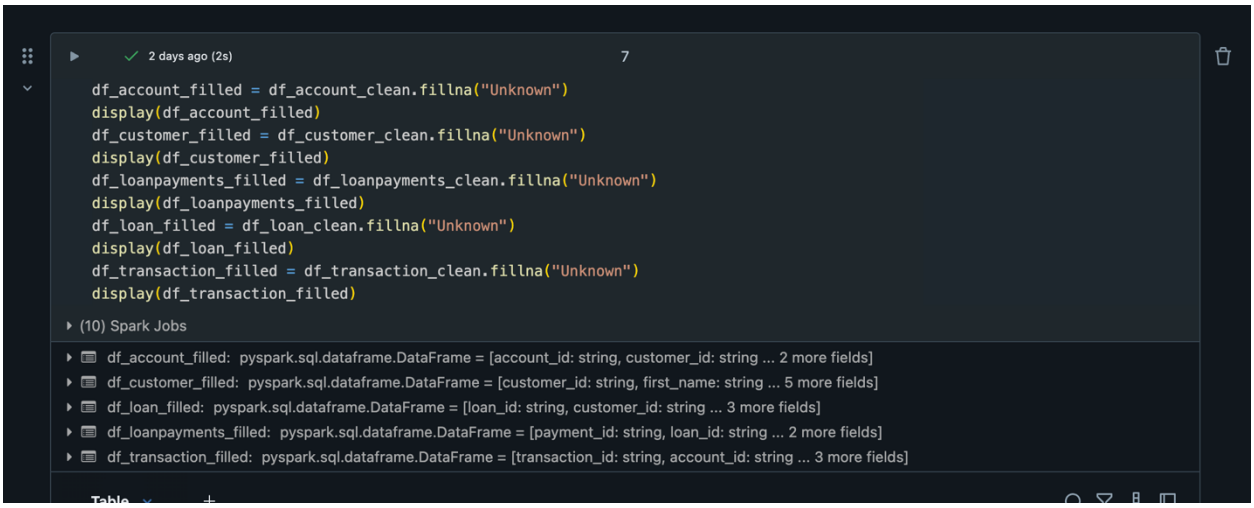
We are checking the secrets are accessible or not

We are configuring our adls to databricks using secrets

We are reading the data in our container



Removing the duplicates



Replacing the null values

2 days ago (13s) 8

```
df_account_filled.write.format("parquet") \
    .mode("overwrite") \
    .save("/mnt/project2/silver/accounts")
df_customer_filled.write.format("parquet") \
    .mode("overwrite") \
    .save("/mnt/project2/silver/customers")
df_loanpayments_filled.write.format("parquet") \
    .mode("overwrite") \
    .save("/mnt/project2/silver/loan_payments")
df_loan_filled.write.format("parquet") \
    .mode("overwrite") \
    .save("/mnt/project2/silver/loans")
df_transaction_filled.write.format("parquet") \
    .mode("overwrite") \
    .save("/mnt/project2/silver/transactions")
```

(10) Spark Jobs

Name	Type	Owner	Created at
Notebook_1	Notebook	Harsha Vardhan	Apr 23, 2025, 04:01 PM
Notebook_2	Notebook	Harsha Vardhan	Apr 23, 2025, 07:40 PM
Notebook_3_goldlayer	Notebook	Harsha Vardhan	Apr 23, 2025, 08:23 PM

Converting the file into parquet format

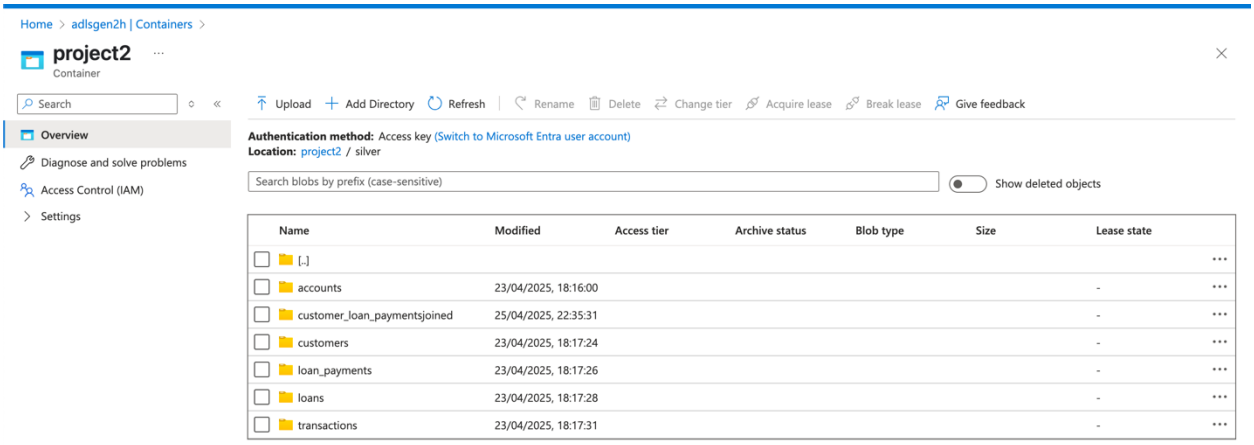
Yesterday (5s) 2

```
from pyspark.sql.functions import col

# Step 1: Join all DataFrames
df_joined = df_accounts_parquet.join(df_transactions_parquet, on="account_id", how="left") \
    .join(df_customers_parquet, on="customer_id", how="left") \
    .join(df_loans_parquet, on="customer_id", how="left") \
    .join(df_loanpayments_parquet, on="loan_id", how="left")

# Step 2: Define grouped order of columns
ordered_columns = [
    # IDs
    "account_id", "transaction_id", "customer_id", "loan_id", "payment_id",

    # Dates grouped
    "transaction_date", "payment_date", "loan_term",

    # Amounts grouped
    "transaction_amount", "payment_amount", "loan_amount", "balance",

    # Customer info grouped
    "first_name", "last_name", "address", "city", "state", "zip"
]

# Step 3: Get any remaining columns
all_columns = df_joined.columns
remaining_columns = [c for c in all_columns if c not in ordered_columns]
```

Now joining all the columns and convert the file into parquet for further visualization and reporting it in fabric

```
10:32 PM (4s)                                                    5
df_clean.write.format("parquet").save("/mnt/project2/silver/customer_loan_paymentsjoined")
(7) Spark Jobs
```

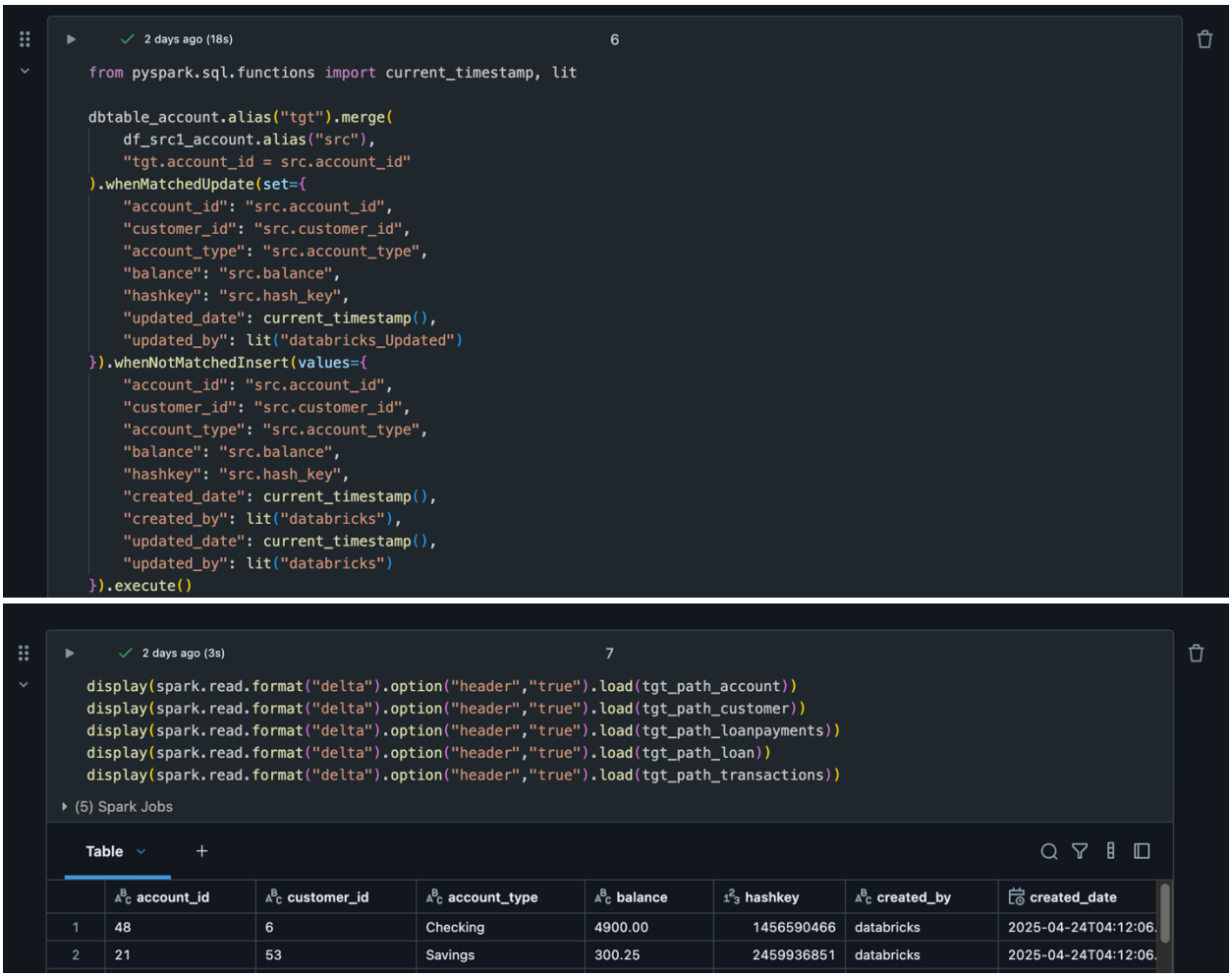
Converted the joined code into parquet format



Files created after running the notebook

```
2 days ago (9s)                                                  1                                                  SQL
%sql
CREATE TABLE if not exists delta.`/mnt/project2/gold/accountsscdtype1` (
  account_id STRING,
  customer_id STRING,
  account_type STRING,
  balance STRING,
 hashkey BIGINT,
  created_by STRING,
  created_date TIMESTAMP,
  updated_by STRING,
  updated_date TIMESTAMP
);

CREATE TABLE if not exists delta.`/mnt/project2/gold/customersscdtype1` (
  customer_id STRING,
  first_name STRING,
  last_name STRING,
  address STRING,
  city STRING,
  state STRING,
  zip STRING,
 hashkey BIGINT,
  created_by STRING,
  created_date TIMESTAMP,
  updated_by STRING,
  updated_date TIMESTAMP
);
```

Applying SCD type 1 transformation to parquet file in silver layer



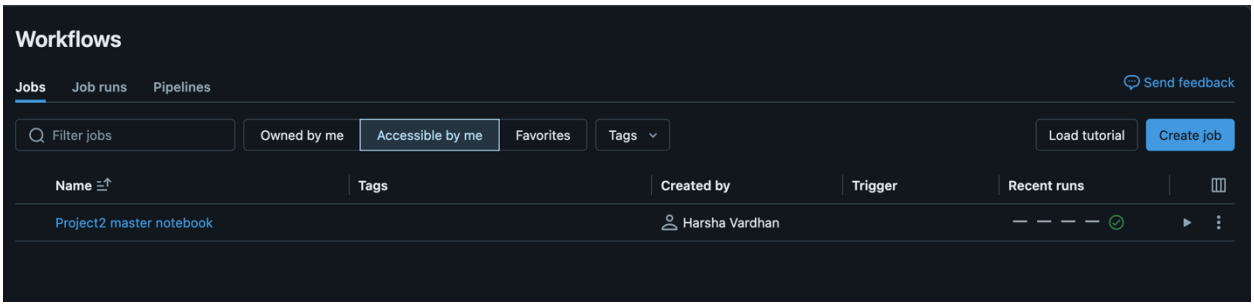
```
2 days ago (18s)                                                    6

from pyspark.sql.functions import current_timestamp, lit

dbtable_account.alias("tgt").merge(
    df_src1_account.alias("src"),
    "tgt.account_id = src.account_id"
).whenMatchedUpdate(set={
    "account_id": "src.account_id",
    "customer_id": "src.customer_id",
    "account_type": "src.account_type",
    "balance": "src.balance",
    "hashkey": "src.hash_key",
    "updated_date": current_timestamp(),
    "updated_by": lit("databricks_Updated")
}).whenNotMatchedInsert(values={
    "account_id": "src.account_id",
    "customer_id": "src.customer_id",
    "account_type": "src.account_type",
    "balance": "src.balance",
    "hashkey": "src.hash_key",
    "created_date": current_timestamp(),
    "created_by": lit("databricks"),
    "updated_date": current_timestamp(),
    "updated_by": lit("databricks")
}).execute()
```

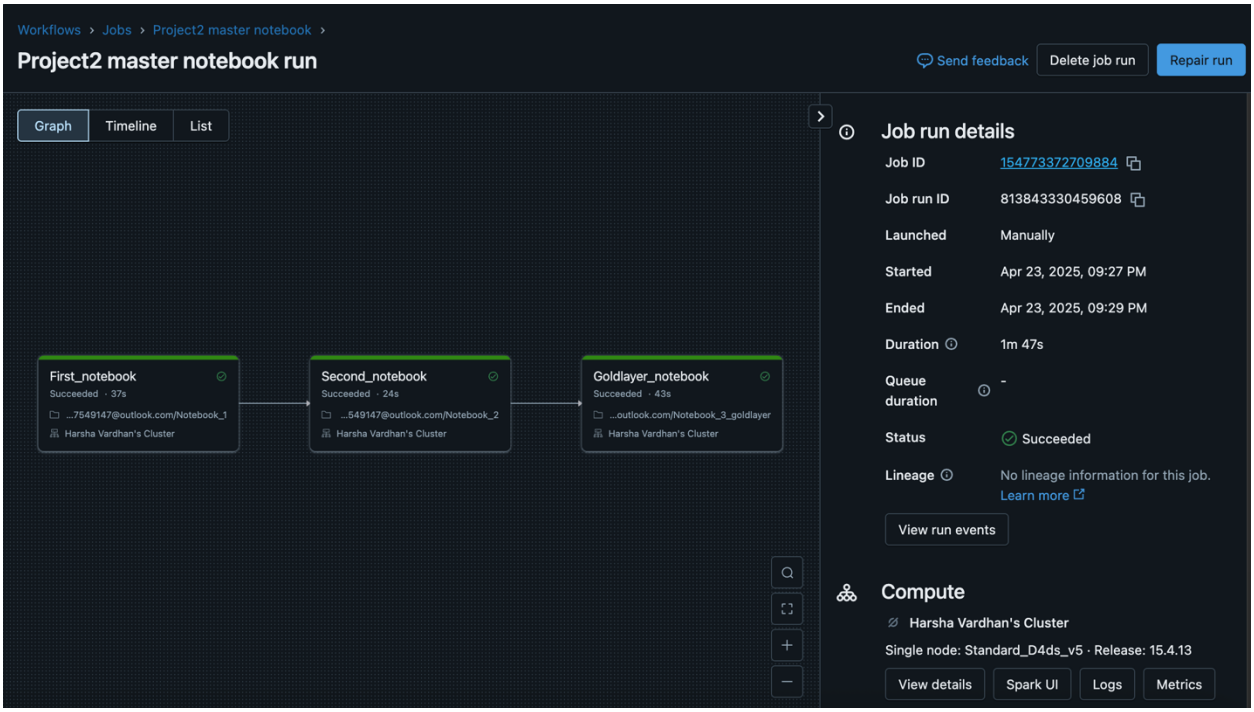
```
2 days ago (3s)                                                     7

display(spark.read.format("delta").option("header","true").load(tgt_path_account))
display(spark.read.format("delta").option("header","true").load(tgt_path_customer))
display(spark.read.format("delta").option("header","true").load(tgt_path_loanpayments))
display(spark.read.format("delta").option("header","true").load(tgt_path_loan))
display(spark.read.format("delta").option("header","true").load(tgt_path_transactions))

▸ (5) Spark Jobs
```

Table +

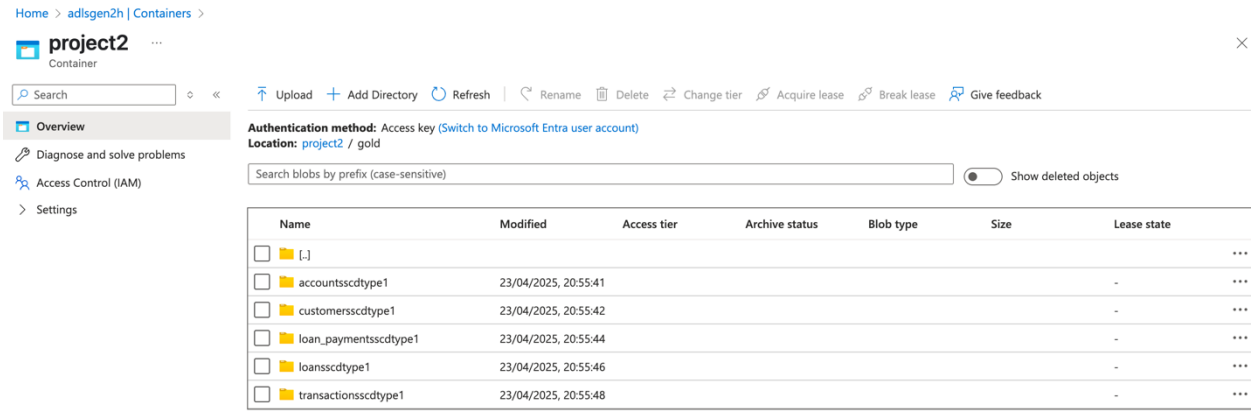
	account_id	customer_id	account_type	balance	hashkey	created_by	created_date
1	48	6	Checking	4900.00	1456590466	databricks	2025-04-24T04:12:06.
2	21	53	Savings	300.25	2459936851	databricks	2025-04-24T04:12:06.

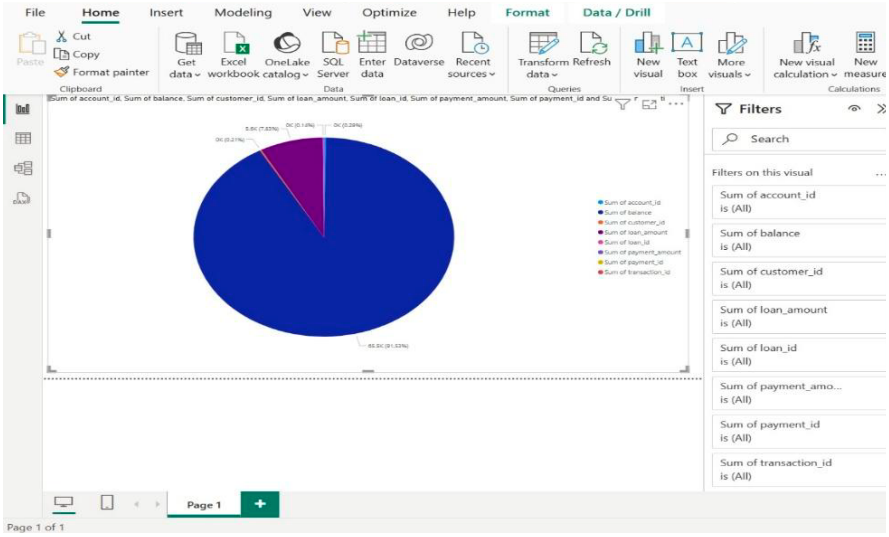
Successfully converted all the files into scd type 1 and stored In gold layer.



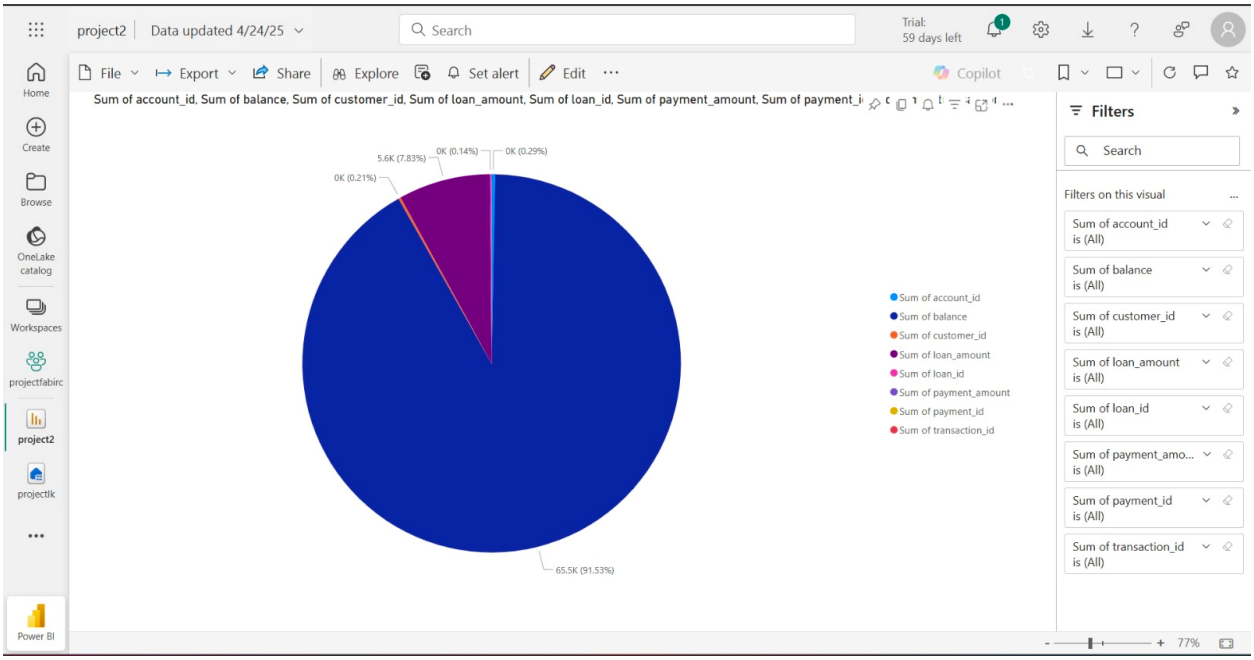


I have created 3 notebook so I created a job to run them all one after the other and it successfully executed.





Visualization created in power bi.



Published in fabric