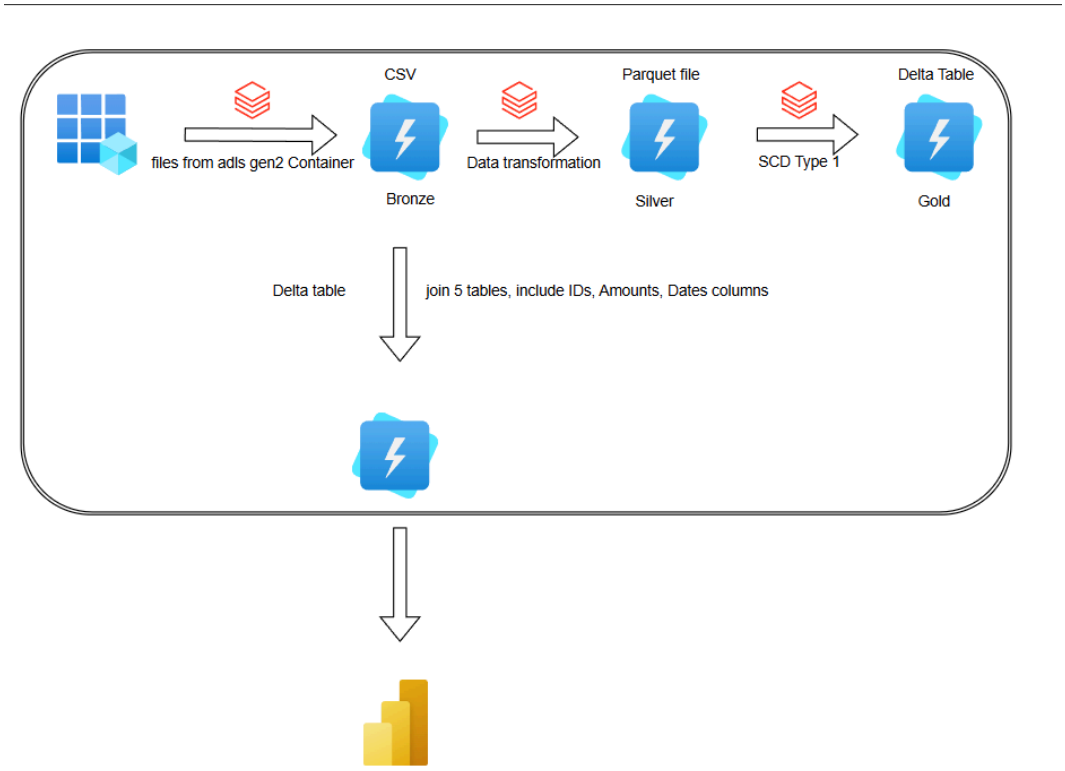
Bootcamp Project 2 - Transactions and Loan Data for a Customer

Introduction

This project uploads five banking CSVs to ADLS Gen2, processes them in Azure Databricks with cleaning, transformation, and SCD Type 1 deduplication. Data is stored in Delta format, secured via Key Vault and mount points, then visualized in Power BI and published to Microsoft Fabric Workspace.

Project Architecture



Architecture of the Project

Implementation

Step 1: Data Ingestion

- Source: Backend team storage account containing files:

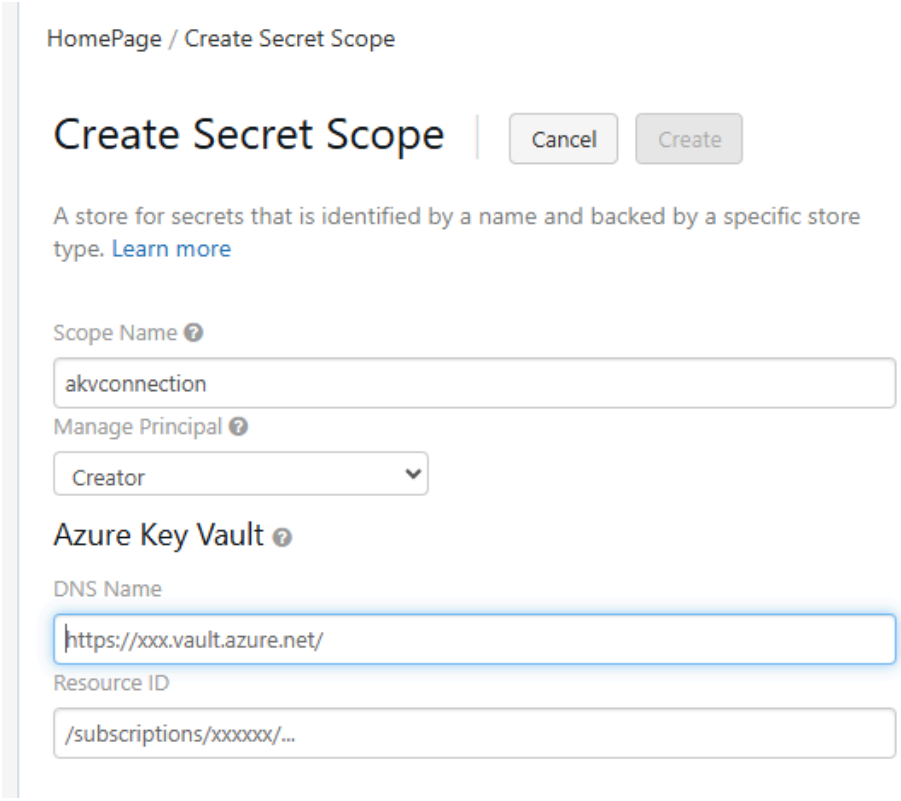
- accounts.csv
- customers.csv
- loan_payments.csv
- loans.csv
- transactions.csv

- Sink: ADLS Gen2 raw (Bronze) container.

List of the files

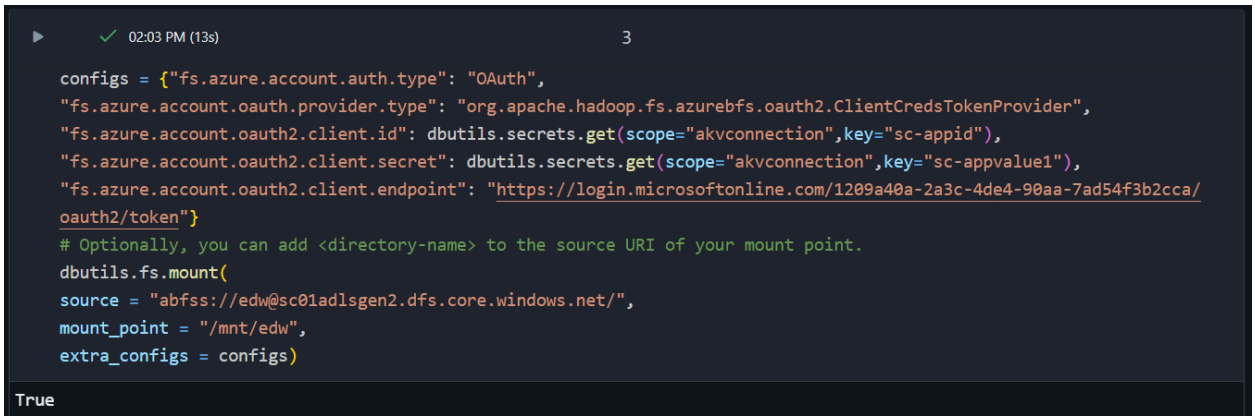
Step 2: Data Cleaning and Transformation

- Environment: Databricks Notebooks
- Mount ADLS Gen2 using scope from Azure Key Vault.



Created the Scope and stored the Connecting Details

- create a mount point to Azure Data Lake gen 2 using below code



- we use pyspark code to read data from bronze layer, remove duplicates and null values and store the cleaned data into Silver layer in parquet format for all the files in Bronze layer.

02:03 PM (14s) 6

```
df_accounts=spark.read.csv("/mnt/edw/bronze/accounts.csv",header=True, inferSchema=True)
df_accounts = df_accounts.na.drop()
df_accounts=df_accounts.dropDuplicates()
df_accounts.write.mode("overwrite").format("parquet") .save("/mnt/edw/silver/Accounts")
display(df_accounts)
```

(6) Spark Jobs

02:04 PM (4s) 7

```
df_customers=spark.read.csv("/mnt/edw/bronze/customers.csv",header=True, inferSchema=True)
df_customers = df_customers.na.drop()
df_customers=df_customers.dropDuplicates()
df_customers.write.mode("overwrite").format("parquet") .save("/mnt/edw/silver/Customers")
display(df_customers)
```

(6) Spark Jobs

df_customers: pyspark.sql.dataframe.DataFrame = [customer_id: integer, first_name: string ... 5 more fields]

02:04 PM (3s) 8

```
df_loan_payments=spark.read.csv("/mnt/edw/bronze/loan_payments.csv",header=True, inferSchema=True)
df_loan_payments = df_loan_payments.na.drop()
df_loan_payments=df_loan_payments.dropDuplicates()
df_loan_payments.write.mode("overwrite").format("parquet") .save("/mnt/edw/silver/Loan_Payment")
display(df_loan_payments)
```

(6) Spark Jobs

df_loan_payments: pyspark.sql.dataframe.DataFrame = [payment_id: integer, loan_id: integer ... 2 more fields]

02:04 PM (3s) 9

```
df_loans=spark.read.csv("/mnt/edw/bronze/loans.csv",header=True, inferSchema=True)
df_loans = df_loans.na.drop()
df_loans=df_loans.dropDuplicates()
df_loans.write.mode("overwrite").format("parquet") .save("/mnt/edw/silver/Loans")
display(df_loans)
```

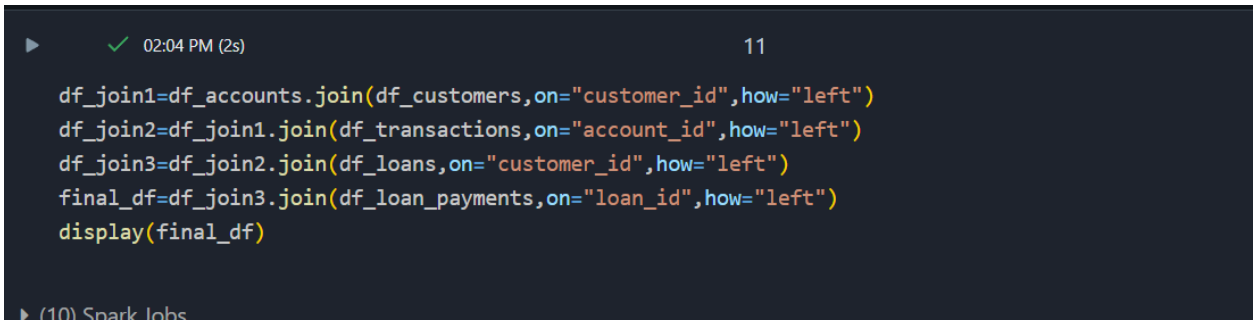
(6) Spark Jobs

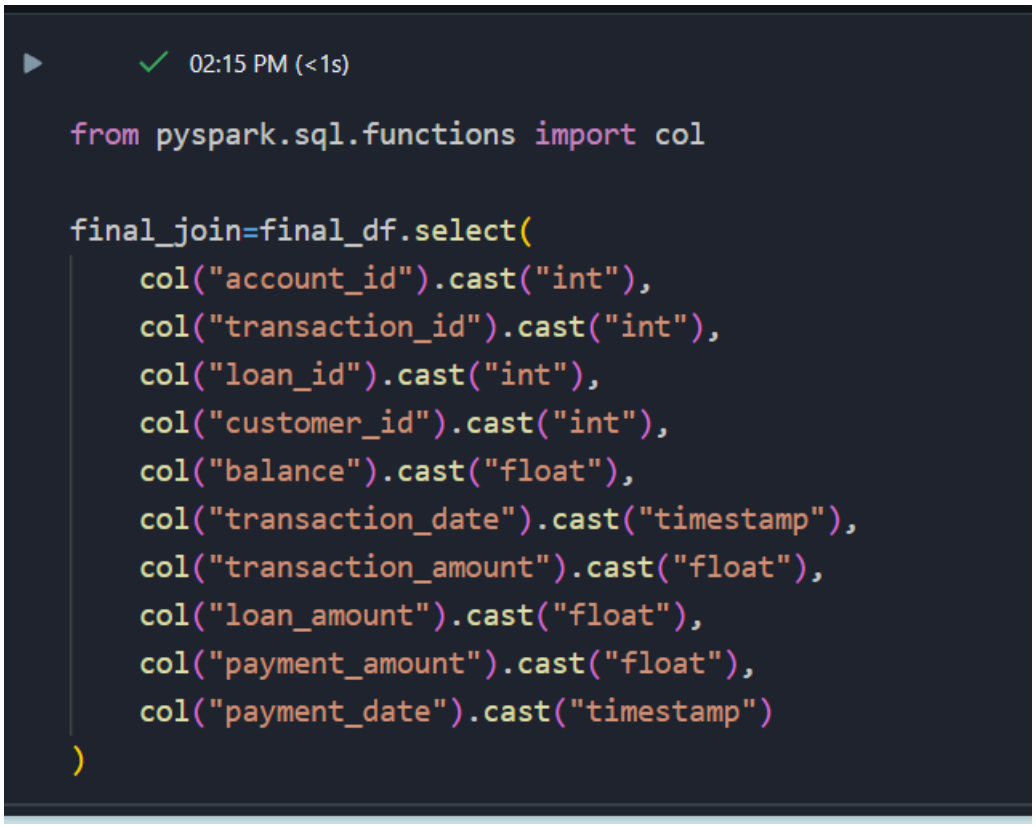
02:04 PM (3s) 10

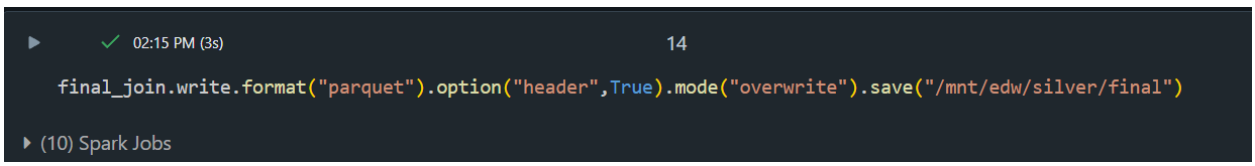
```
df_transactions=spark.read.csv("/mnt/edw/bronze/transactions.csv",header=True, inferSchema=True)
df_transactions = df_transactions.na.drop()
df_transactions=df_transactions.dropDuplicates()
df_transactions.write.mode("overwrite").format("parquet") .save("/mnt/edw/silver/Transactions")
display(df_transactions)
```

(6) Spark Jobs

- Join all these files into one file and store in Silver later in parquet format.







- perform SCD Type 1 on all the files data.

02:26 PM (10s) 15

```
%sql

CREATE TABLE IF NOT EXISTS delta.`/mnt/edw/gold/Accounts/` (
  account_id int,
  customer_id int,
  account_type string,
  balance float,
  hash_key bigint,
  created_by string,
  created_date timestamp,
  updated_by string,
  updtaed_date timestamp
)
```

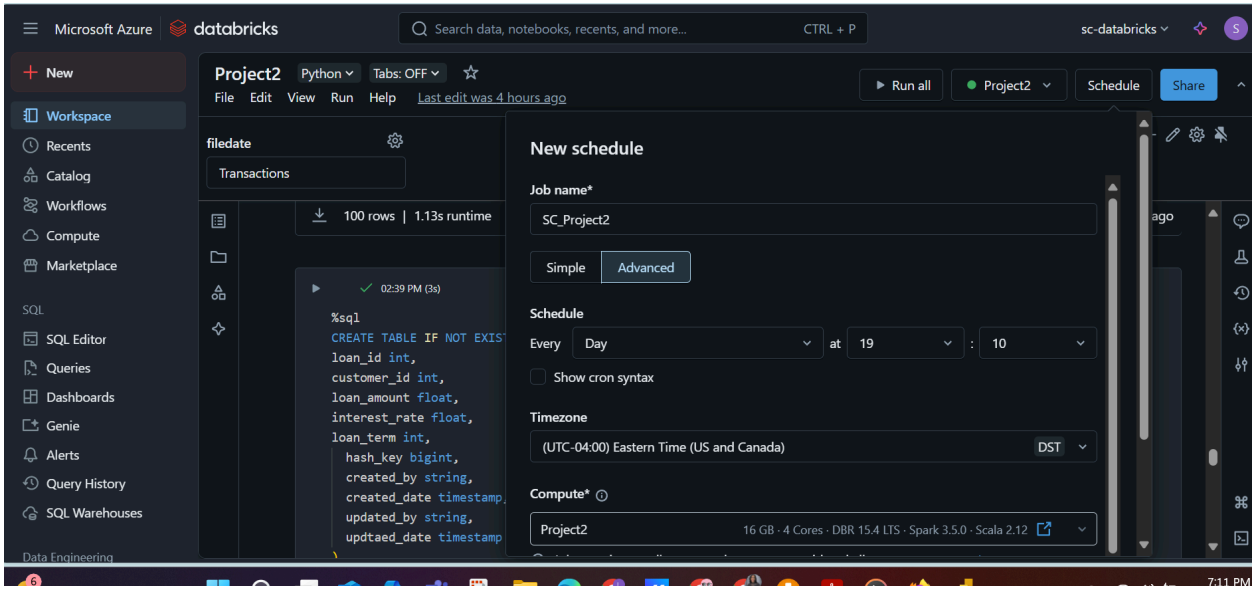
02:41 PM (<1s) 16

```
from pyspark.sql.functions import  *
dbutils.widgets.text("filedate","")
filedate=dbutils.widgets.get("filedate")
```

02:32 PM (2s) 17

```
src_path="/mnt/edw/silver/"+filedate
print(src_path)
tgt_path="/mnt/edw/gold/Accounts/"
print(tgt_path)
df_src=spark.read.format("parquet").option("header","true").option("inferSchema","true").load(src_path)
display(df_src)
df_src1=df_src.withColumn("hash_key",crc32(concat(*df_src.columns)))
```

Once SCD Type 1 is done we have to create a schedule job to run pipeline everyday or any specified time


Microsoft Azure
databricks
Search data, notebooks, recents, and more...
CTRL + P
sc-databricks
New
Workspace
Recents
Catalog
Workflows
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Project2
Python
Tabs: OFF
File Edit View Run Help
Last edit was 4 hours ago
Run all
Project2
Schedule
Share
filedate
Transactions
100 rows | 1.13s runtime
02:39 PM (3s)
%sql
CREATE TABLE IF NOT EXIS
loan_id int,
customer_id int,
loan_amount float,
interest_rate float,
loan_term int,
hash_key bigint,
created_by string,
created_date timestamp,
updated_by string,
updtaed_date timestamp
New schedule
Job name*
SC_Project2
Simple
Advanced
Schedule
Every
Day
at
19
:
10
Show cron syntax
Timezone
(UTC-04:00) Eastern Time (US and Canada)
DST
Compute*
Project2
16 GB · 4 Cores · DBR 15.4 LTS · Spark 3.5.0 · Scala 2.12
7:11 PM