

DSCI 417 – Homework 08

Sean Graham

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, expr

spark = SparkSession.builder.getOrCreate()
```

Problem 1: Creating Streaming DataFrame

```
static_df = (spark.read.option('header', True).option('inferSchema', True).csv('/FileStore/tables/paysim/step_001.csv'))

paysim_schema = static_df.schema

static_df.show(5)
```

step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest
1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0
1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0
1	TRANSFER	181.0	C1305486145	181.0	0.0	C553264065	0.0	0.0
1	CASH_OUT	181.0	C840083671	181.0	0.0	C38997010	21182.0	0.0
1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0

only showing top 5 rows

```
paysim_stream = (  
    spark.readStream  
        .option('header', True)  
        .option('maxFilesPerTrigger', 1)  
        .schema(paysim_schema)  
        .csv('/FileStore/tables/paysim/')  
)
```

```
print(paysim_stream.isStreaming)
```

```
True
```

Problem 2: Apply Transformations

```
type_summary = (  
    paysim_stream  
        .groupBy('type')  
        .agg(  
            expr('COUNT(*) AS n'),  
            expr('ROUND(MEAN(amount),2) AS avg_amount'),  
            expr('MIN(amount) AS min_amount'),  
            expr('MAX(amount) AS max_amount')  
        )  
        .sort('n', ascending=False)  
)
```

```
destinations = (  
    paysim_stream  
    .filter(expr('type == "TRANSFER"'))  
    .groupBy('nameDest')  
    .agg(  
        expr('COUNT(*) AS n'),  
        expr('SUM(amount) AS total_amount'),  
        expr('ROUND(MEAN(amount),2) AS avg_amount')  
    )  
    .sort('n', ascending=False)  
)
```



Problem 3: Define Output Sinks

```
type_summary_object = (  
    type_summary  
    .writeStream  
    .format('memory')  
    .queryName('type_summary')  
    .outputMode('complete')  
)
```

```
destinations_object = (  
    destinations  
    .writeStream  
    .format('memory')  
    .queryName('destinations')  
    .outputMode('complete')  
)
```

Problem 4: Start and Monitor the Streams

```
type_query = type_symmary_object.start()
dest_query = destinations_object.start()
```

- ▶  type_summary (id: 40df665c-81ef-4c19-a73d-b4ce24474920) Last updated: 1,352 days ago
- ▶  destinations (id: ecd66813-edc1-4864-b405-366ad4ddb0bb) Last updated: 1,352 days ago

```
type_summary_static = spark.sql('select * from type_summary')
print(type_summary_static.count())
type_summary_static.show()
```

5

type	n	avg_amount	min_amount	max_amount
CASH_OUT	263202	186676.06	0.37	1.0E7
PAYMENT	245255	11716.55	0.1	115264.68
CASH_IN	159806	172214.77	5.44	1289407.91
TRANSFER	60447	673269.92	2.6	1.0E7
DEBIT	5115	5946.25	0.87	218148.28

```
destinations_static = spark.sql('select * from destinations')
print(destinations_static.count())
destinations_static.show(16)
```

28668

+-----+-----+-----+-----+			
nameDest	n	total_amount	avg_amount
+-----+-----+-----+-----+			
C665576141	26	2.6150103900000002E7	1005773.23
C1286084959	26	1.7588259099999998E7	676471.5
C248609774	24	2.4711862799999997E7	1029660.95
C1360767589	22	2.1756362919999994E7	988925.59
C306206744	22	1.3048885069999998E7	593131.14
C985934102	21	1.3709050239999998E7	652811.92
C451111351	20	1.9783480340000004E7	989174.02
C1590550415	20	2.6350395919999994E7	1317519.8
C97730845	20	2.8009878859999996E7	1400493.94
C716083600	19	2.026890477E7	1066784.46
C1883840933	19	2.386476521E7	1256040.27
C2083562754	18	2.198921771E7	1221623.21
C863811613	18	2.1868866990000002E7	1214937.06
C803116137	17	1.8642155009999998E7	1096597.35
C2011996123	16	1.459595107E7	912246.94
C1255024717	16	1.4511803379999999E7	906987.71

