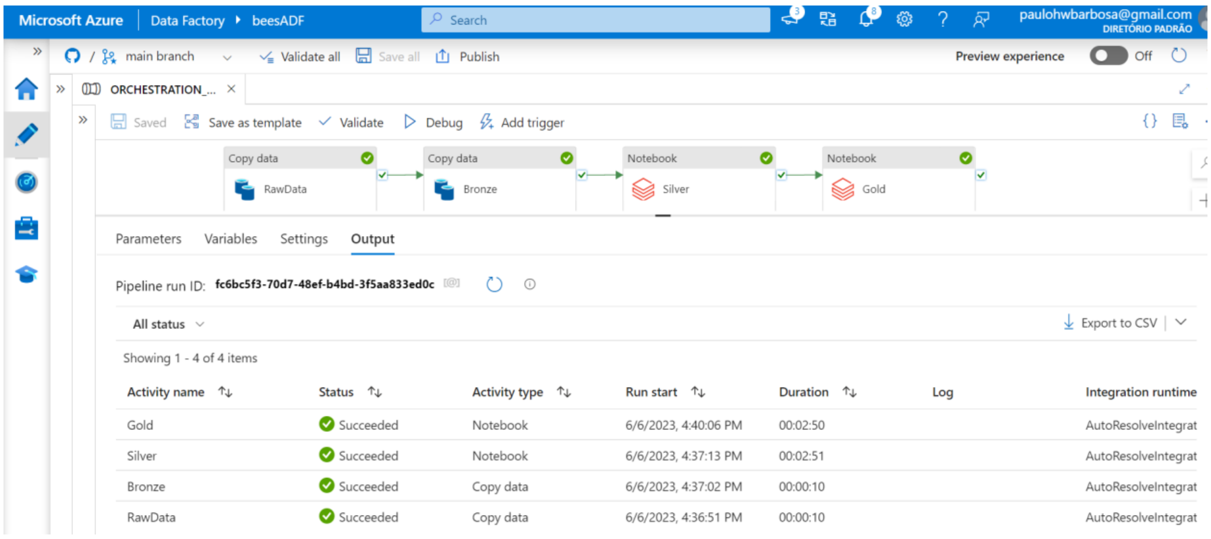
Azure Data Factory

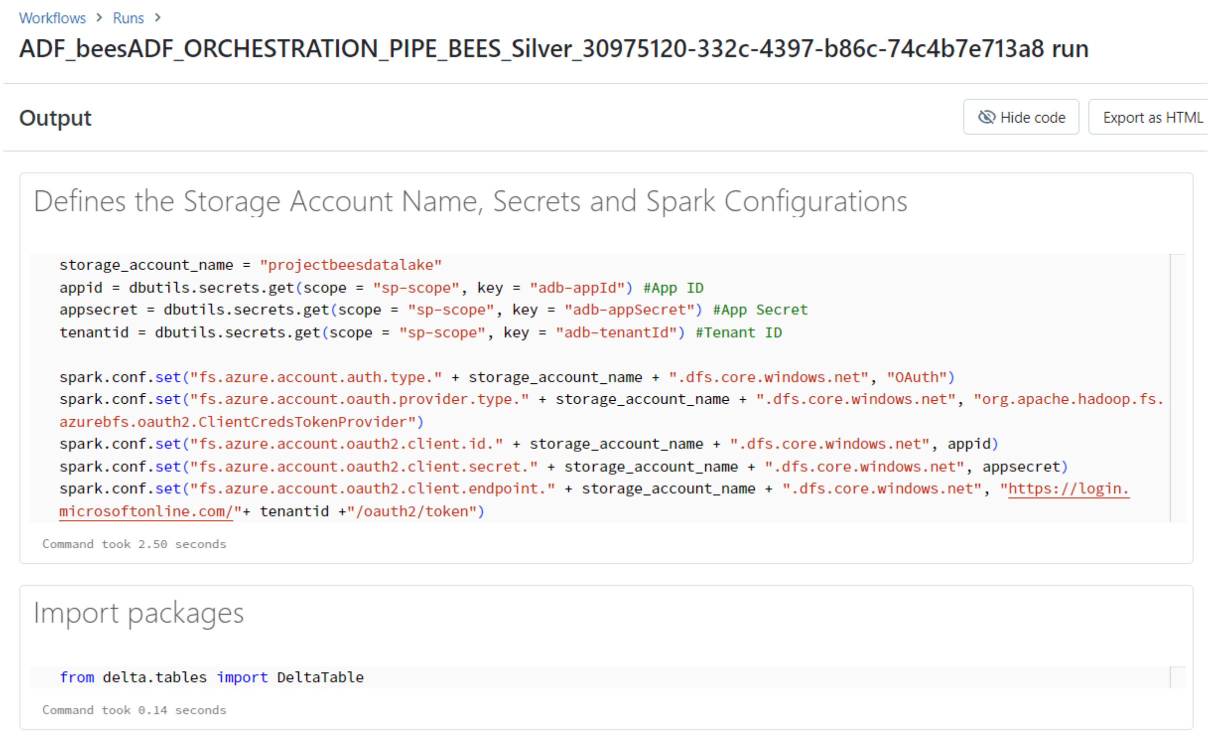


Pipeline run ID: fc6bc5f3-70d7-48ef-b4bd-3f5aa833ed0c

Showing 1 - 4 of 4 items

Activity name	Status	Activity type	Run start	Duration	Log	Integration runtime
Gold	Succeeded	Notebook	6/6/2023, 4:40:06 PM	00:02:50		AutoResolveIntegrat
Silver	Succeeded	Notebook	6/6/2023, 4:37:13 PM	00:02:51		AutoResolveIntegrat
Bronze	Succeeded	Copy data	6/6/2023, 4:37:02 PM	00:00:10		AutoResolveIntegrat
RawData	Succeeded	Copy data	6/6/2023, 4:36:51 PM	00:00:10		AutoResolveIntegrat

Databricks Silver Notebook



ADF_beesADF_ORCHESTRATION_PIPE_BEES_Silver_30975120-332c-4397-b86c-74c4b7e713a8 run

Output

Defines the Storage Account Name, Secrets and Spark Configurations

```
storage_account_name = "projectbeesdatalake"
appid = dbutils.secrets.get(scope = "sp-scope", key = "adb-appId") #App ID
appsecret = dbutils.secrets.get(scope = "sp-scope", key = "adb-appSecret") #App Secret
tenantid = dbutils.secrets.get(scope = "sp-scope", key = "adb-tenantId") #Tenant ID

spark.conf.set("fs.azure.account.auth.type." + storage_account_name + ".dfs.core.windows.net", "OAuth")
spark.conf.set("fs.azure.account.oauth.provider.type." + storage_account_name + ".dfs.core.windows.net", "org.apache.hadoop.fs.
azurebfs.oauth2.ClientCredsTokenProvider")
spark.conf.set("fs.azure.account.oauth2.client.id." + storage_account_name + ".dfs.core.windows.net", appid)
spark.conf.set("fs.azure.account.oauth2.client.secret." + storage_account_name + ".dfs.core.windows.net", appsecret)
spark.conf.set("fs.azure.account.oauth2.client.endpoint." + storage_account_name + ".dfs.core.windows.net", "https://login.
microsoftonline.com/"+ tenantid +"/oauth2/token")
```

```
Command took 2.50 seconds
```

Import packages

```
from delta.tables import DeltaTable
```

```
Command took 0.14 seconds
```

Defines containers location

```python
silver_storage_container = "silver"

adls_silver_base_path = f"abfss://{silver_storage_container}" \
                        f"@{storage_account_name}.dfs.core.windows.net"

job_base_path_silver = f"{adls_silver_base_path}/datalake/json/"

job_input_path = f"{job_base_path_silver}/input/breweries.json"
job_staging_path = f"{job_base_path_silver}/staging/"
job_output_path = f"{job_base_path_silver}/data/"
```

```
Command took 0.08 seconds
```

Create the schema

```sql
%sql

CREATE SCHEMA IF NOT EXISTS bees
```

```
▸ _sqldf: pyspark.sql.dataframe.DataFrame
OK
Command took 8.10 seconds
```

Read JSON data

```python
df_input = spark.read.json(job_input_path)
```

```
▸ df_input: pyspark.sql.dataframe.DataFrame = [address_1: string, address_2: string ... 14 more fields]
Command took 9.25 seconds
```

Create delta table partitioned by location

```python
df_input.write.saveAsTable("bees.beweries_silver",
                           format="delta",
                           mode="overwrite",
                           path=job_output_path,
                           partitionBy="state")
```

```
Command took 25.37 seconds
```

Saving at the staging directory

```python
df_input.write.mode('overwrite').format('delta').save(job_staging_path)
```

```
Command took 3.41 seconds
```

Applies Vaccum

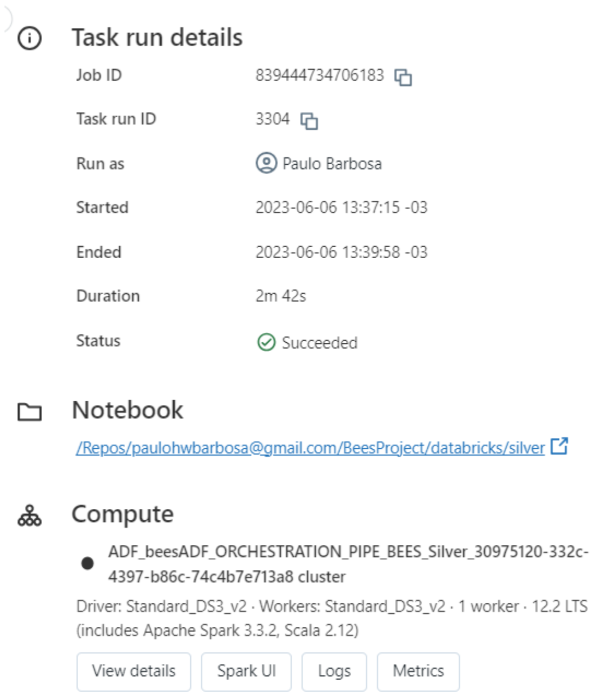
```python
delta_table = DeltaTable.forName(spark, "bees.beweries_silver")
delta_table.vacuum()
```

```
Out[8]: DataFrame[]
Command took 19.61 seconds
```

Delete data from input

```python
try:
    dbutils.fs.rm(job_input_path, True)
except Exception as e:
    raise Exception(f'Error removing data from {job_input_path}.\
                Exception: {e}')
```

```
Command took 0.19 seconds
```



Task run details

Job ID	839444734706183
Task run ID	3304
Run as	Paulo Barbosa
Started	2023-06-06 13:37:15 -03
Ended	2023-06-06 13:39:58 -03
Duration	2m 42s
Status	Succeeded

Notebook

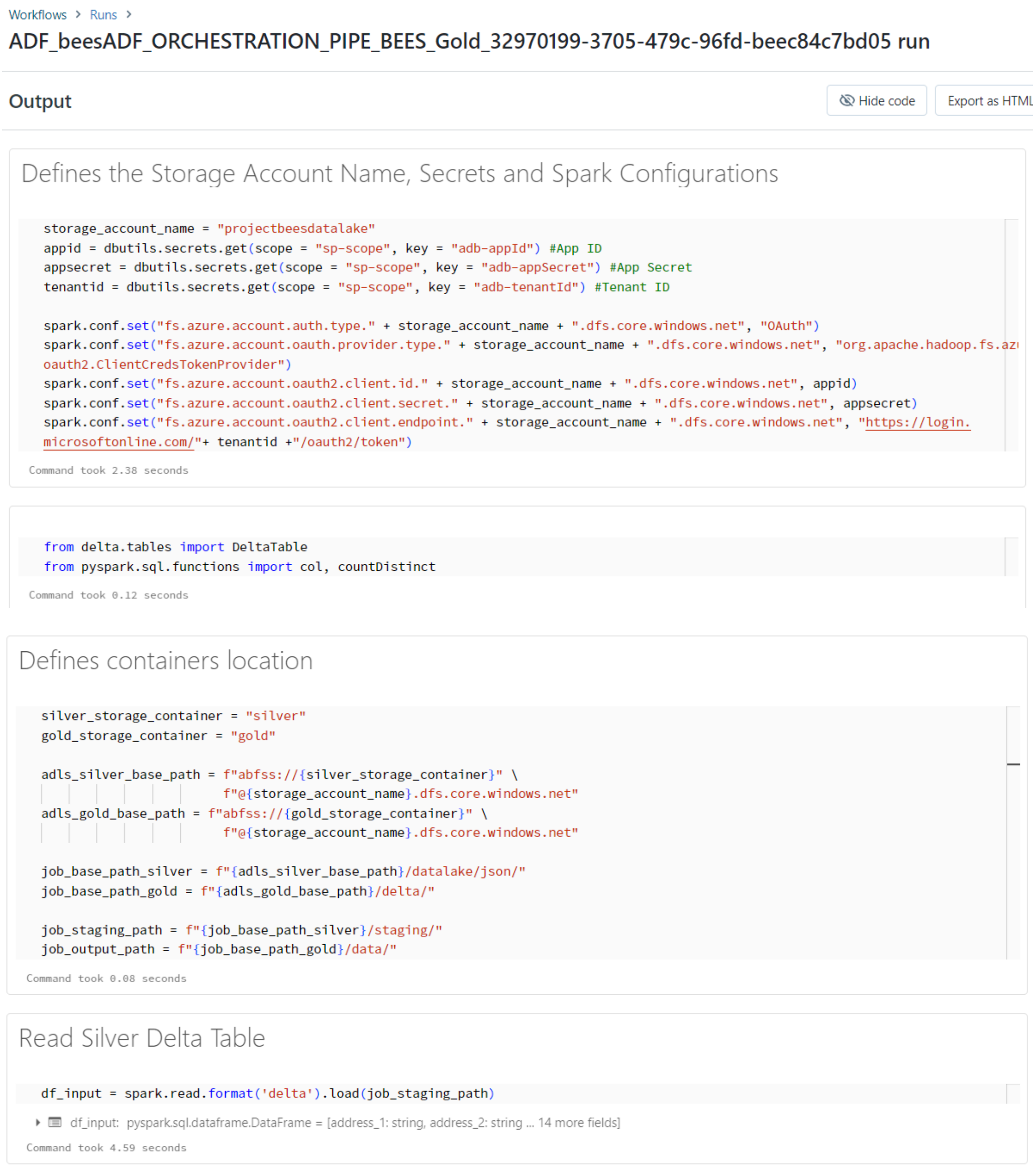
/Repos/paulohwbarbosa@gmail.com/BeesProject/databricks/silver

Compute

● ADF_beesADF_ORCHESTRATION_PIPE_BEES_Silver_30975120-332c-4397-b86c-74c4b7e713a8 cluster

Driver: Standard_DS3_v2 · Workers: Standard_DS3_v2 · 1 worker · 12.2 LTS (includes Apache Spark 3.3.2, Scala 2.12)

Databricks Gold Notebook



Workflows › Runs ›

ADF_beesADF_ORCHESTRATION_PIPE_BEES_Gold_32970199-3705-479c-96fd-beec84c7bd05 run

Output

Defines the Storage Account Name, Secrets and Spark Configurations

```
storage_account_name = "projectbeesdatalake"
appid = dbutils.secrets.get(scope = "sp-scope", key = "adb-appId") #App ID
appsecret = dbutils.secrets.get(scope = "sp-scope", key = "adb-appSecret") #App Secret
tenantid = dbutils.secrets.get(scope = "sp-scope", key = "adb-tenantId") #Tenant ID

spark.conf.set("fs.azure.account.auth.type." + storage_account_name + ".dfs.core.windows.net", "OAuth")
spark.conf.set("fs.azure.account.oauth.provider.type." + storage_account_name + ".dfs.core.windows.net", "org.apache.hadoop.fs.az
oauth2.ClientCredsTokenProvider")
spark.conf.set("fs.azure.account.oauth2.client.id." + storage_account_name + ".dfs.core.windows.net", appid)
spark.conf.set("fs.azure.account.oauth2.client.secret." + storage_account_name + ".dfs.core.windows.net", appsecret)
spark.conf.set("fs.azure.account.oauth2.client.endpoint." + storage_account_name + ".dfs.core.windows.net", "https://login.
microsoftonline.com/"+ tenantid +"/oauth2/token")
```

Command took 2.38 seconds

```
from delta.tables import DeltaTable
from pyspark.sql.functions import col, countDistinct
```

Command took 0.12 seconds

Defines containers location

```
silver_storage_container = "silver"
gold_storage_container = "gold"

adls_silver_base_path = f"abfss://{silver_storage_container}" \
                        f"@{storage_account_name}.dfs.core.windows.net"
adls_gold_base_path = f"abfss://{gold_storage_container}" \
                        f"@{storage_account_name}.dfs.core.windows.net"

job_base_path_silver = f"{adls_silver_base_path}/datalake/json/"
job_base_path_gold = f"{adls_gold_base_path}/delta/"

job_staging_path = f"{job_base_path_silver}/staging/"
job_output_path = f"{job_base_path_gold}/data/"
```

Command took 0.08 seconds

Read Silver Delta Table

```
df_input = spark.read.format('delta').load(job_staging_path)
```

▸ df_input: pyspark.sql.dataframe.DataFrame = [address_1: string, address_2: string ... 14 more fields]

Command took 4.59 seconds

Select only the necessary columns

```
df_beweries = df_input.select(col("id").alias("id"),
                        col("name").alias("name"),
                        col("brewery_type").alias("brewery_type"),
                        col("city").alias("city"),
                        col("state").alias("state"),
                        col("country").alias("country"))
```

▸ df_beweries: pyspark.sql.dataframe.DataFrame = [id: string, name: string ... 4 more fields]

```
Command took 0.29 seconds
```

Create aggregated views

```
df_aggregate_type = df_beweries.groupBy("brewery_type").agg(countDistinct("id")).createOrReplaceTempView("BreweryTypeCount")
df_aggregate_location = df_beweries.groupBy("country", "state", "city").agg(countDistinct("id")).createOrReplaceTempView
("BreweryLocationCount")
```

```
Command took 2.56 seconds
```

```
%sql
SELECT * FROM BreweryTypeCount
```

▾ _sqldf: pyspark.sql.dataframe.DataFrame

```
brewery_type: string
count(id): long
```

Table

	brewery_type	count(id)
1	brewpub	9
2	proprietor	1
3	contract	1
4	closed	2
5	micro	30
6	large	7

6 rows | 21.91 seconds runtime

```
Command took 21.91 seconds
```

```
%sql
SELECT * FROM BreweryLocationCount
```

▾ _sqldf: pyspark.sql.dataframe.DataFrame

```
country: string
state: string
city: string
count(id): long
```

Table

	country	state	city	count(id)
1	United States	Indiana	Gary	1
2	United States	California	San Diego	1
3	United States	Illinois	Assumption	1
4	United States	Iowa	Des Moines	1
5	United States	Michigan	Jackson	1
6	United States	Delaware	Georgetown	1
7	Ireland	Laois	Killeshin	1

46 rows | 1.59 seconds runtime

```
Command took 1.59 seconds
```

Create delta table partitioned by location

```
df_beweries.write.saveAsTable("bees.beweries_gold",
                              format="delta",
                              mode="overwrite",
                              path=job_output_path,
                              partitionBy="state")
```

```
Command took 18.55 seconds
```

Applies Vaccum

```
delta_table = DeltaTable.forName(spark, "bees.beweries_gold")
delta_table.vacuum()
```

```
Out[10]: DataFrame[]
Command took 18.99 seconds
```

Delete staging from silver layer

```
try:
    dbutils.fs.rm(job_staging_path, True)
except Exception as e:
    raise Exception(f'Error removing data from {job_staging_path}.\
                Exception: {e}')
```

```
Command took 0.23 seconds
```

Task run details

Job ID	645424486273634
Task run ID	3975
Run as	Paulo Barbosa
Started	2023-06-06 13:40:07 -03
Ended	2023-06-06 13:42:39 -03
Duration	2m 31s
Status	✓ Succeeded

Notebook

/Repos/paulohwbarbosa@gmail.com/BeesProject/databricks/gold

Compute

● ADF_beesADF_ORCHESTRATION_PIPE_BEES_Gold_32970199-3705-479c-96fd-beec84c7bd05 cluster

Driver: Standard_DS3_v2 · Workers: Standard_DS3_v2 · 1 worker · 12.2 LTS (includes Apache Spark 3.3.2, Scala 2.12)

View details | Spark UI | Logs | Metrics