Spark Table - Temp vs Permanent

February 4, 2025

[1]:
```
from pyspark.sql import SparkSession

# Initialize SparkSession with Hive support
spark = SparkSession.builder \
    .appName("TableDemo") \
    .enableHiveSupport() \
    .getOrCreate()

# Read CSV from HDFS
hdfs_path = "/tmp/customers_100.csv"  # Update path as per your HDFS location
df = spark.read.option("header", True).csv(hdfs_path)

# Show Data
df.show(5)
```

```
25/02/02 03:09:38 WARN SparkSession: Using an existing Spark session; only
runtime SQL configurations will take effect.
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|customer_id|      name|     city|
state|country|registration_date|is_active|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|          0|Customer_0|     Pune|Maharashtra|  India|      2023-06-29|
False|
|          1|Customer_1|Bangalore| Tamil Nadu|  India|      2023-12-07|
True|
|          2|Customer_2|Hyderabad|    Gujarat|  India|      2023-10-27|
True|
|          3|Customer_3|Bangalore|  Karnataka|  India|      2023-10-17|
False|
|          4|Customer_4|Ahmedabad|  Karnataka|  India|      2023-03-14|
False|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
only showing top 5 rows
```

[18]:
```
spark.sql('show tables').show(truncate=False)
```

```
+---------+-------------+-----------+
|namespace|tableName    |isTemporary|
+---------+-------------+-----------+
|         |temp_customers|true       |
+---------+-------------+-----------+
```

[13]:
```
spark.sql('drop table temp_customers_2').show()
```

```
++
||
++
++
```

[15]:
```
# ---------------- Step 1: Create a Temporary Table (Session-based)␣
↪---------------- #
df.createOrReplaceTempView("temp_customers")

print("### Querying Temporary Table (Exists only in this session) ###")
spark.sql("SELECT * FROM temp_customers LIMIT 5").show()
```

```
### Querying Temporary Table (Exists only in this session) ###
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|customer_id|      name|     city|
state|country|registration_date|is_active|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|          0|Customer_0|     Pune|Maharashtra|  India|      2023-06-29|
False|
|          1|Customer_1|Bangalore| Tamil Nadu|  India|      2023-12-07|
True|
|          2|Customer_2|Hyderabad|    Gujarat|  India|      2023-10-27|
True|
|          3|Customer_3|Bangalore|  Karnataka|  India|      2023-10-17|
False|
|          4|Customer_4|Ahmedabad|  Karnataka|  India|      2023-03-14|
False|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
```

[17]:
```
# ---------------- Step 2: Create a Global Temporary Table (Accessible across␣
↪sessions) ---------------- #
df.createOrReplaceGlobalTempView("global_customers")
```

```
print("### Querying Global Temporary Table ###")
spark.sql("SELECT * FROM global_temp.global_customers LIMIT 5").show()
```

```
### Querying Global Temporary Table ###
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|customer_id|      name|     city|
state|country|registration_date|is_active|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|          0|Customer_0|     Pune|Maharashtra|  India|      2023-06-29|
False|
|          1|Customer_1|Bangalore| Tamil Nadu|  India|      2023-12-07|
True|
|          2|Customer_2|Hyderabad|    Gujarat|  India|      2023-10-27|
True|
|          3|Customer_3|Bangalore|  Karnataka|  India|      2023-10-17|
False|
|          4|Customer_4|Ahmedabad|  Karnataka|  India|      2023-03-14|
False|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
```

[19]:
```
spark.sql('select * from global_temp.global_customers limit 5').show()
```

```
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|customer_id|      name|     city|
state|country|registration_date|is_active|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|          0|Customer_0|     Pune|Maharashtra|  India|      2023-06-29|
False|
|          1|Customer_1|Bangalore| Tamil Nadu|  India|      2023-12-07|
True|
|          2|Customer_2|Hyderabad|    Gujarat|  India|      2023-10-27|
True|
|          3|Customer_3|Bangalore|  Karnataka|  India|      2023-10-17|
False|
|          4|Customer_4|Ahmedabad|  Karnataka|  India|      2023-03-14|
False|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
```

[20]:
```
# ---------------- Step 3: Create a Persistent Table (Stored in Hive Metastore)␣
  ↪---------------- #
spark.sql("DROP TABLE IF EXISTS customers_persistent")
df.write.mode("overwrite").saveAsTable("customers_persistent")

print("### Querying Persistent Table (Accessible across sessions and␣
  ↪applications) ###")
spark.sql("SELECT * FROM customers_persistent LIMIT 5").show()
```

```
25/02/02 03:16:05 WARN SessionState: METASTORE_FILTER_HOOK will be ignored,
since hive.security.authorization.manager is set to instance of
HiveAuthorizerFactory.

### Querying Persistent Table (Accessible across sessions and applications) ###
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|customer_id|      name|     city|
state|country|registration_date|is_active|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
|          0|Customer_0|     Pune|Maharashtra|  India|      2023-06-29|
False|
|          1|Customer_1|Bangalore| Tamil Nadu|  India|      2023-12-07|
True|
|          2|Customer_2|Hyderabad|    Gujarat|  India|      2023-10-27|
True|
|          3|Customer_3|Bangalore|  Karnataka|  India|      2023-10-17|
False|
|          4|Customer_4|Ahmedabad|  Karnataka|  India|      2023-03-14|
False|
+-----------+----------+---------+-----------+-------+----------------+--------
-+
```

[23]:
```
spark.sql('describe extended customers_persistent').show(truncate =False)
```

```
+----------------------------+----------------------------------------------------
----------+-------+
|col_name                    |data_type
|comment|
+----------------------------+----------------------------------------------------
----------+-------+
|customer_id                 |string
|null   |
|name                        |string
|null   |
|city                        |string
|null   |
|state                       |string
```

```
|null   |
|country                    |string
|null   |
|registration_date          |string
|null   |
|is_active                  |string
|null   |
|                           |
|       |
|# Detailed Table Information|
|       |
|Database                   |default
|       |
|Table                      |customers_persistent
|       |
|Owner                      |root
|       |
|Created Time               |Sun Feb 02 03:16:06 UTC 2025
|       |
|Last Access                |UNKNOWN
|       |
|Created By                 |Spark 3.3.2
|       |
|Type                       |MANAGED
|       |
|Provider                   |parquet
|       |
|Statistics                 |3898 bytes
|       |
|Location                   |hdfs://my-
cluster-m/user/hive/warehouse/customers_persistent|       |
|Serde Library
|org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe |       |
+---------------------------+--------------------------------------------------
----------+-------+
only showing top 20 rows
```

[28]:
```python
# Create a new session within the same Spark application
spark_new = spark.newSession()

# Verify the view exists
print("### Querying Global Temporary Table ###")
spark_new.sql("SHOW TABLES IN global_temp").show()

# Query the Global Temp View
spark_new.sql("SELECT * FROM global_temp.global_customers").show()
```

```
### Querying Global Temporary Table ###
+-----------+---------------+-----------+
|  namespace|      tableName|isTemporary|
+-----------+---------------+-----------+
|global_temp|global_customers|       true|
+-----------+---------------+-----------+

+-----------+-----------+---------+-----------+-------+----------------+-------
--+
|customer_id|       name|     city|
state|country|registration_date|is_active|
+-----------+-----------+---------+-----------+-------+----------------+-------
--+
|          0| Customer_0|     Pune|Maharashtra|  India|      2023-06-29|
False|
|          1| Customer_1|Bangalore| Tamil Nadu|  India|      2023-12-07|
True|
|          2| Customer_2|Hyderabad|    Gujarat|  India|      2023-10-27|
True|
|          3| Customer_3|Bangalore|  Karnataka|  India|      2023-10-17|
False|
|          4| Customer_4|Ahmedabad|  Karnataka|  India|      2023-03-14|
False|
|          5| Customer_5|Hyderabad|  Karnataka|  India|      2023-07-28|
False|
|          6| Customer_6|     Pune|      Delhi|  India|      2023-08-29|
False|
|          7| Customer_7|Ahmedabad|West Bengal|  India|      2023-12-28|
True|
|          8| Customer_8|     Pune|  Karnataka|  India|      2023-06-22|
True|
|          9| Customer_9|   Mumbai|  Telangana|  India|      2023-01-05|
True|
|         10|Customer_10|     Pune|    Gujarat|  India|      2023-08-05|
True|
|         11|Customer_11|    Delhi|West Bengal|  India|      2023-08-02|
False|
|         12|Customer_12|  Chennai|    Gujarat|  India|      2023-11-21|
False|
|         13|Customer_13|  Chennai|  Karnataka|  India|      2023-11-06|
True|
|         14|Customer_14|Hyderabad| Tamil Nadu|  India|      2023-02-07|
False|
|         15|Customer_15|   Mumbai|    Gujarat|  India|      2023-03-02|
True|
|         16|Customer_16|  Chennai|  Karnataka|  India|      2023-04-05|
False|
|         17|Customer_17|Hyderabad|West Bengal|  India|      2023-08-21|
```

```
False|
|         18|Customer_18|     Pune|      Delhi|  India|       2023-10-04|
True|
|         19|Customer_19|  Kolkata|    Gujarat|  India|       2023-02-05|
True|
+-----------+-----------+---------+-----------+-------+-----------------+-------
--+
only showing top 20 rows
```

[29]:
```
spark_new.sql("SELECT * FROM temp_customers LIMIT 5").show()
```

```
---------------------------------------------------------------------------
AnalysisException                         Traceback (most recent call last)
/tmp/ipykernel_9027/193810830.py in <cell line: 1>()
----> 1 spark_new.sql("SELECT * FROM temp_customers LIMIT 5").show()

/usr/lib/spark/python/pyspark/sql/session.py in sql(self, sqlQuery, **kwargs)
   1032             sqlQuery = formatter.format(sqlQuery, **kwargs)
   1033         try:
-> 1034             return DataFrame(self._jsparkSession.sql(sqlQuery), self)
   1035         finally:
   1036             if len(kwargs) > 0:

/usr/lib/spark/python/lib/py4j-0.10.9.5-src.zip/py4j/java_gateway.py in␣
 ↪__call__(self, *args)
   1319
   1320         answer = self.gateway_client.send_command(command)
-> 1321         return_value = get_return_value(
   1322             answer, self.gateway_client, self.target_id, self.name)
   1323

/usr/lib/spark/python/pyspark/sql/utils.py in deco(*a, **kw)
    194                 # Hide where the exception came from that shows a␣
 ↪non-Pythonic
    195                 # JVM exception message.
--> 196                 raise converted from None
    197             else:
    198                 raise

AnalysisException: Table or view not found: temp_customers; line 1 pos 14;
'GlobalLimit 5
+- 'LocalLimit 5
   +- 'Project [*]
      +- 'UnresolvedRelation [temp_customers], [], false
```

[32]:
```
spark.sql('show tables').show()
```

```
+---------+--------------------+-----------+
|namespace|           tableName|isTemporary|
+---------+--------------------+-----------+
|  default|customers_persistent|      false|
|         |      temp_customers|       true|
+---------+--------------------+-----------+
```

[33]:
```
spark.stop()
```