Module 3 - Data Integration and Aggregation

March 7, 2025

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

spark = SparkSession.builder \
.appName('OlistData') \
.getOrCreate()
```

```
25/02/28 09:03:58 WARN SparkSession: Using an existing Spark session; only
runtime SQL configurations will take effect.
```

[2]:
```python
hdfs_path = '/olist/'
```

[3]:
```python
customers_df = spark.read.csv(hdfs_path + 'olist_customers_dataset.
 ↪csv',header=True,inferSchema=True)
orders_df = spark.read.csv(hdfs_path + 'olist_orders_dataset.
 ↪csv',header=True,inferSchema=True)
order_item_df = spark.read.csv(hdfs_path + 'olist_order_items_dataset.
 ↪csv',header=True,inferSchema=True)
payments_df = spark.read.csv(hdfs_path + 'olist_order_payments_dataset.
 ↪csv',header=True,inferSchema=True)
reviews_df = spark.read.csv(hdfs_path + 'olist_order_reviews_dataset.
 ↪csv',header=True,inferSchema=True)
products_df = spark.read.csv(hdfs_path + 'olist_products_dataset.
 ↪csv',header=True,inferSchema=True)
sellers_df = spark.read.csv(hdfs_path + 'olist_sellers_dataset.
 ↪csv',header=True,inferSchema=True)
geolocation_df = spark.read.csv(hdfs_path + 'olist_geolocation_dataset.
 ↪csv',header=True,inferSchema=True)
category_translation_df = spark.read.csv(hdfs_path +␣
 ↪'product_category_name_translation.csv',header=True,inferSchema=True)
```

[4]:
```python
# Cache Frequently used Data for Better Performance

orders_df.cache()
customers_df.cache()
order_item_df.cache()
```

```
[4]: DataFrame[order_id: string, order_item_id: int, product_id: string, seller_id:
     string, shipping_limit_date: timestamp, price: double, freight_value: double]
```

```
[ ]:
```

```
[5]: orders_items_joined_df = orders_df.join(order_item_df,'order_id','inner')
```

```
[6]: orders_items_products_df = orders_items_joined_df.
      ↪join(products_df,'product_id','inner')
```

```
[7]: orders_items_products_sellers_df = orders_items_products_df.
      ↪join(sellers_df,'seller_id','inner')
```

```
[8]: full_orders_df = orders_items_products_sellers_df.
      ↪join(customers_df,'customer_id','inner')
```

```
[9]: # GEolocation Data

     full_orders_df = full_orders_df.join(geolocation_df,full_orders_df.
      ↪customer_zip_code_prefix == geolocation_df.
      ↪geolocation_zip_code_prefix,'left')
```

```
[10]: full_orders_df = full_orders_df.join(reviews_df,'order_id','left')
```

```
[11]: full_orders_df = full_orders_df.join(payments_df,'order_id','left')
```

```
[12]: full_orders_df.cache()
```

```
25/02/28 08:37:30 WARN package: Truncated the string representation of a plan
since it was too large. This behavior can be adjusted by setting
'spark.sql.debug.maxToStringFields'.
```

```
[12]: DataFrame[order_id: string, customer_id: string, seller_id: string, product_id:
      string, order_status: string, order_purchase_timestamp: timestamp,
      order_approved_at: timestamp, order_delivered_carrier_date: timestamp,
      order_delivered_customer_date: timestamp, order_estimated_delivery_date:
      timestamp, order_item_id: int, shipping_limit_date: timestamp, price: double,
      freight_value: double, product_category_name: string, product_name_lenght: int,
      product_description_lenght: int, product_photos_qty: int, product_weight_g: int,
      product_length_cm: int, product_height_cm: int, product_width_cm: int,
      seller_zip_code_prefix: int, seller_city: string, seller_state: string,
      customer_unique_id: string, customer_zip_code_prefix: int, customer_city:
      string, customer_state: string, geolocation_zip_code_prefix: int,
      geolocation_lat: double, geolocation_lng: double, geolocation_city: string,
      geolocation_state: string, review_id: string, review_score: string,
      review_comment_title: string, review_comment_message: string,
      review_creation_date: string, review_answer_timestamp: string,
      payment_sequential: int, payment_type: string, payment_installments: int,
 payment_value: double]
```

[18]:
```
from pyspark.sql.functions import *
```

[14]:
```
full_orders_df.printSchema()
```

```
root
 |-- order_id: string (nullable = true)
 |-- customer_id: string (nullable = true)
 |-- seller_id: string (nullable = true)
 |-- product_id: string (nullable = true)
 |-- order_status: string (nullable = true)
 |-- order_purchase_timestamp: timestamp (nullable = true)
 |-- order_approved_at: timestamp (nullable = true)
 |-- order_delivered_carrier_date: timestamp (nullable = true)
 |-- order_delivered_customer_date: timestamp (nullable = true)
 |-- order_estimated_delivery_date: timestamp (nullable = true)
 |-- order_item_id: integer (nullable = true)
 |-- shipping_limit_date: timestamp (nullable = true)
 |-- price: double (nullable = true)
 |-- freight_value: double (nullable = true)
 |-- product_category_name: string (nullable = true)
 |-- product_name_lenght: integer (nullable = true)
 |-- product_description_lenght: integer (nullable = true)
 |-- product_photos_qty: integer (nullable = true)
 |-- product_weight_g: integer (nullable = true)
 |-- product_length_cm: integer (nullable = true)
 |-- product_height_cm: integer (nullable = true)
 |-- product_width_cm: integer (nullable = true)
 |-- seller_zip_code_prefix: integer (nullable = true)
 |-- seller_city: string (nullable = true)
 |-- seller_state: string (nullable = true)
 |-- customer_unique_id: string (nullable = true)
 |-- customer_zip_code_prefix: integer (nullable = true)
 |-- customer_city: string (nullable = true)
 |-- customer_state: string (nullable = true)
 |-- geolocation_zip_code_prefix: integer (nullable = true)
 |-- geolocation_lat: double (nullable = true)
 |-- geolocation_lng: double (nullable = true)
 |-- geolocation_city: string (nullable = true)
 |-- geolocation_state: string (nullable = true)
 |-- review_id: string (nullable = true)
 |-- review_score: string (nullable = true)
 |-- review_comment_title: string (nullable = true)
 |-- review_comment_message: string (nullable = true)
 |-- review_creation_date: string (nullable = true)
 |-- review_answer_timestamp: string (nullable = true)
 |-- payment_sequential: integer (nullable = true)
 |-- payment_type: string (nullable = true)
 |-- payment_installments: integer (nullable = true)
 |-- payment_value: double (nullable = true)
```

[19]:
```python
# Total Revenues Per Seller


seller_revenue_df = full_orders_df.groupBy('seller_id').agg(sum('price'))
```

[20]:
```python
seller_revenue_df.show(5)
```

```
[Stage 26:=====================================================>(199 + 1) / 200]

+--------------------+------------------+
|           seller_id|        sum(price)|
+--------------------+------------------+
|e63e8bfa530fb1691…|219481.00000000035|
|ff063b022a9a0aab9…|         1860394.0|
|a49928bcdf77c55c6…|1220624.6000000054|
|33ac3e28642ab8bda…| 615628.8499999995|
|7aa4334be125fcdd2…| 2509294.489999999|
+--------------------+------------------+
only showing top 5 rows
```

[ ]:
```python

```

[ ]:
```python
# Total Orders Per Custoemr
# Average Review Score Per Seller
# Most Sold Products ( Top 10 )
# Top Custoemrs By Spending
```

[ ]:
```python

```

1 Optimized Joins For Data integration

[8]:
```python
from pyspark.sql.functions import *
```

[5]:
```python
orders_items_joined_df = orders_df.join(order_item_df,'order_id','inner')
```

[6]:
```python
orders_items_products_df = orders_items_joined_df.
  ↪join(products_df,'product_id','inner')
```

[9]:
```python
orders_items_products_sellers_df = orders_items_products_df.
  ↪join(broadcast(sellers_df),'seller_id','inner')
```

[10]:
```
full_orders_df = orders_items_products_sellers_df.
  ↪join(customers_df,'customer_id','inner')
```

[11]:
```
# GEolocation Data

full_orders_df = full_orders_df.join(broadcast(geolocation_df),full_orders_df.
  ↪customer_zip_code_prefix == geolocation_df.
  ↪geolocation_zip_code_prefix,'left')
```

[12]:
```
full_orders_df = full_orders_df.join(broadcast(reviews_df),'order_id','left')
```

[13]:
```
full_orders_df = full_orders_df.join(payments_df,'order_id','left')
```

[15]:
```
full_orders_df.cache()
```

```
25/02/28 09:07:07 WARN package: Truncated the string representation of a plan
since it was too large. This behavior can be adjusted by setting
'spark.sql.debug.maxToStringFields'.
```

[15]:
```
DataFrame[order_id: string, customer_id: string, seller_id: string, product_id:
string, order_status: string, order_purchase_timestamp: timestamp,
order_approved_at: timestamp, order_delivered_carrier_date: timestamp,
order_delivered_customer_date: timestamp, order_estimated_delivery_date:
timestamp, order_item_id: int, shipping_limit_date: timestamp, price: double,
freight_value: double, product_category_name: string, product_name_lenght: int,
product_description_lenght: int, product_photos_qty: int, product_weight_g: int,
product_length_cm: int, product_height_cm: int, product_width_cm: int,
seller_zip_code_prefix: int, seller_city: string, seller_state: string,
customer_unique_id: string, customer_zip_code_prefix: int, customer_city:
string, customer_state: string, geolocation_zip_code_prefix: int,
geolocation_lat: double, geolocation_lng: double, geolocation_city: string,
geolocation_state: string, review_id: string, review_score: string,
review_comment_title: string, review_comment_message: string,
review_creation_date: string, review_answer_timestamp: string,
payment_sequential: int, payment_type: string, payment_installments: int,
payment_value: double]
```

2 Aggregation

[18]:
```
# Total Orders Per Customer

customer_order_count_df = full_orders_df.groupBy('customer_id')\
.agg(count('order_id').alias('total_orders'))\
.orderBy(desc('total_orders'))

customer_order_count_df.show(5)
```

```
[Stage 25:=============================>                              (1 + 1) / 2]
```

```
+--------------------+------------+
|         customer_id|total_orders|
+--------------------+------------+
|351e40989da90e704...|       11427|
|50920f8cd0681fd86...|       10752|
|9b43e2a62de9bab3a...|        8556|
|270c23a11d024a44c...|        8001|
|d3e82ccec3cb5f956...|        6876|
+--------------------+------------+
only showing top 5 rows
```

[20]:
```
#Average Review Score Per Seller

seller_review_df = full_orders_df.groupBy('seller_id')\
.agg(avg('review_score').alias('avg_review_score'))\
.orderBy(desc('avg_review_score'))

seller_review_df.show()
```

```
[Stage 28:=============================>                          (1 + 1) / 2]

+--------------------+----------------+
|           seller_id|avg_review_score|
+--------------------+----------------+
|a353b1083c9863d75...|             5.0|
|bd43e172d599bed47...|             5.0|
|a61cc04793308395a...|             5.0|
|1f2eebc0e970fd3c4...|             5.0|
|7ad41305e96a6cab8...|             5.0|
|64c9a1db4e73e19aa...|             5.0|
|89757206b887aed36...|             5.0|
|a56a8043ebf66e421...|             5.0|
|f1fdf2d1318657575...|             5.0|
|929f342384a6607af...|             5.0|
|05a48cc8859962767...|             5.0|
|7238f877570096ae4...|             5.0|
|1cd9e0cc1839d5551...|             5.0|
|b5b800c4065bebf4d...|             5.0|
|392f7f2c797e4dc07...|             5.0|
|05ca864204d09595a...|             5.0|
|94d76e96eedd97625...|             5.0|
|9d213f303afae4983...|             5.0|
|d598f929fc44e1e38...|             5.0|
|0ad80de75c8113263...|             5.0|
+--------------------+----------------+
only showing top 20 rows
```

[21]:
```python
# Top 10 Most Sold Products

top_products_df = full_orders_df.groupBy('product_id')\
.agg(count('order_id').alias('total_sold'))\
.orderBy(desc('total_sold'))\
.limit(10)

top_products_df.show()
```

```
[Stage 31:=============================>                            (1 + 1) / 2]

+--------------------+----------+
|          product_id|total_sold|
+--------------------+----------+
|aca2eb7d00ea1a7b8…|     86740|
|422879e10f4668299…|     81110|
|99a4788cb24856965…|     78775|
|389d119b48cf3043d…|     60248|
|d1c427060a0f73f6b…|     59274|
|368c6c730842d7801…|     58358|
|53759a2ecddad2bb8…|     52654|
|53b36df67ebb7c415…|     52105|
|154e7e31ebfa09220…|     42700|
|3dd2a17168ec895c7…|     40787|
+--------------------+----------+
```

[ ]:
```python
# Top 10 Customer By Spending
```

[ ]:
```python

```

3 Window Function and Ranking

[23]:
```python
from pyspark.sql.window import Window
```

[24]:
```python
# Dense Rank for Sellers Based on Revenue

window_spec = Window.partitionBy('seller_id').orderBy(desc('price'))
```

[26]:
```python
# Rank Top Selling Products Per seller

top_seller_products_df = full_orders_df.withColumn('rank',rank().
 ↪over(window_spec)).filter(col('rank')<=5)
```

```
top_seller_products_df.select('seller_id','price','rank').show()
```

```
[Stage 39:>                                                         (0 + 1) / 1]

+--------------------+-----+----+
|           seller_id|price|rank|
+--------------------+-----+----+
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
|0015a82c2db000af6...|895.0|   1|
+--------------------+-----+----+
only showing top 20 rows
```

```
[ ]:
```

```
[ ]: # Dense Rank for Sellers Based on Revenue
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```

4 Advance Aggregation and Enrichment

[ ]:
```
full_orders_df
```

[28]:
```
# Total Revenue & Average Order Value (AOV) per Customer

customer_spending_df = full_orders_df.groupBy('customer_id')\
.agg(
    count('order_id').alias('total_orders'),
    sum('price').alias('total_spent'),
    round(avg('price'),2).alias('AOV')
)\
.orderBy(desc('total_spent'))


customer_spending_df.show()
```

```
[Stage 40:=============================>                              (1 + 1) / 2]

+--------------------+------------+------------------+-------+
|         customer_id|total_orders|       total_spent|    AOV|
+--------------------+------------+------------------+-------+
|d3e82ccec3cb5f956…|        6876|         6662844.0|  969.0|
|df55c14d1476a9a34…|         743|         3565657.0| 4799.0|
|fe5113a38e3575c04…|        2292|         3293604.0| 1437.0|
|ec5b2ba62e5743423…|        1428|         2556120.0| 1790.0|
|63b964e79dee32a35…|        6072|         2501664.0|  412.0|
|46bb3c0b1a65c8399…|         748|         2336752.0| 3124.0|
|05455dfa7cd02f13d…|        2184| 2160194.400000087|  989.1|
|3690e975641f01bd0…|         802|         2124498.0| 2649.0|
|349509b216bd5ec11…|         743|         1923627.0| 2589.0|
|695476b5848d64ba0…|         687|1820543.1299999943|2649.99|
|73236a0796f53d60d…|         832|         1755520.0| 2110.0|
|cc803a2c412833101…|         762|         1676400.0| 2200.0|
|1ff773612ab8934db…|        5820|1658641.7999999512| 284.99|
|fced842c7dad61e8c…|         602|         1654898.0| 2749.0|
|1ecb47d23dc8203cd…|        1164|1629588.3599999903|1399.99|
|de832e8dbb1f588a4…|        2190|1584990.5999999817| 723.74|
|803cd9b04f9cd252c…|         488|         1512312.0| 3099.0|
|d72181923840c8895…|        2721|1488114.8999999566|  546.9|
|06d478ba352a27a51…|        1146|         1461150.0| 1275.0|
|0049e8442c2a3e4a8…|        1204|         1444800.0| 1200.0|
+--------------------+------------+------------------+-------+
only showing top 20 rows
```

[29]:
```
6662844/6876
```

[29]: 969.0

[ ]:

[31]:
```
# Seller Performance Metrics ( Revenue, Average Review, Order Count)

seller_performance_df = full_orders_df.groupBy('seller_id') \
.agg(
    count('order_id').alias('total_orders'),
    sum('price').alias('total_revenue'),
    round(avg('review_score'),2).alias('avg_review_score'),
    round(stddev('price'),2).alias('price_variability')
)\
.orderBy(desc('total_revenue'))
```

[32]:
```
seller_performance_df.show()
```

```
[Stage 43:==============================>                          (1 + 1) / 2]

+--------------------+------------+--------------------+----------------+-------
----------+
|           seller_id|total_orders|
total_revenue|avg_review_score|price_variability|
+--------------------+------------+--------------------+----------------+-------
----------+
|4869f7a5dfa277a7d…|      184587|3.6138717319998816E7|            4.09|
111.65|
|53243585a1d6dc264…|       54514| 3.429159295000016E7|            4.12|
499.65|
|4a3ca9315b744ce9f…|      330661| 3.375957084003399E7|            3.77|
59.37|
|7c67e1448b00f6e96…|      233306|3.2282321790014144E7|            3.42|
50.39|
|fa1c13f2614d7b5c4…|       87686| 3.013938631000357E7|            4.38|
307.7|
|da8622b14eb17ae28…|      264433|  2.98576697300434E7|            3.98|
72.92|
|7e93a43ef30c4f03f…|       50226| 2.631570630000493E7|            4.15|
377.24|
|1025f0e2d44d7041d…|      229587|2.2937518520012498E7|            3.89|
84.3|
|46dc3b2cc0980fb8e…|       90426| 2.179177329001596E7|            4.18|
187.49|
|955fee9216a65b617…|      232364|2.0964410670014285E7|            4.04|
84.94|
|7a67c85e85bb2ce85…|      167231|2.0312794890029624E7|            4.26|
56.23|
|620c87c171fb2a6dd…|      142232| 2.011983960002556E7|            4.36|
100.45|
|7d13fca1522535862…|       88807| 1.815688191000456E7|            4.07|
151.18|
|a1043bafd471dff53…|      132672|1.7662675980011847E7|            4.25|
37.19|
|6560211a19b47992c…|      286539|1.7315932900000416E7|            3.86|
35.04|
|edb1ef5e36e0c8cd8…|       38945|1.6624835150005734E7|            4.43|
460.85|
|1f50f920176fa81da…|      297292|1.6497454440035844E7|            4.04|
7.39|
|5dceca129747e92ff…|       50420|1.4910548340005763E7|            4.17|
299.84|
|cc419e0650a3c5ba7…|      256032|1.4751464500039315E7|            4.07|
22.67|
|3d871de0142ce09b7…|      175876|1.4184525300005388E7|            4.15|
38.14|
+--------------------+------------+--------------------+----------------+-------
----------+
only showing top 20 rows
```

[34]:
```
# Product Popularity Metrics

product_metrics_df = full_orders_df.groupBy('product_id')\
.agg(
    count('order_id').alias('total_sales'),
    sum('price').alias('total_revenue'),
    round(avg('price'),2).alias('avg_price'),
    round(stddev('price'),2).alias('price_volatility'),\
    collect_set('seller_id').alias('unique_sellers')
)\
.orderBy(desc('total_sales'))
```

[35]:
```
product_metrics_df.show()
```

```
[Stage 46:=============================>                              (1 + 1) / 2]

+--------------------+-----------+------------------+---------+----------------
+--------------------+
|          product_id|total_sales|     total_revenue|avg_price|price_volatility|
unique_sellers|
+--------------------+-----------+------------------+---------+----------------
+--------------------+
|aca2eb7d00ea1a7b8…|      86740|6164630.2999962345|    71.07|
3.17|[955fee9216a65b61…|
|422879e10f4668299…|      81110| 4442791.509997333|    54.77|
4.46|[1f50f920176fa81d…|
|99a4788cb24856965…|      78775| 6921762.709995905|    87.87|
4.08|[4a3ca9315b744ce9…|
|389d119b48cf3043d…|      60248| 3280533.129998912|    54.45|
4.37|[1f50f920176fa81d…|
|d1c427060a0f73f6b…|      59274| 8220103.330002628|   138.68|
16.58|[a1043bafd471dff5…|
|368c6c730842d7801…|      58358| 3181698.899999065|    54.52|
4.59|[1f50f920176fa81d…|
|53759a2ecddad2bb8…|      52654| 2893017.499999481|    54.94|
4.52|[1f50f920176fa81d…|
|53b36df67ebb7c415…|      52105| 6159887.409998229|   118.22|
20.13|[7d13fca152253586…|
|154e7e31ebfa09220…|      42700| 962160.9999997382|    22.53|
1.92|[cc419e0650a3c5ba…|
|3dd2a17168ec895c7…|      40787| 6116941.299997734|   149.97|
0.85|[de722cd6dad950a9…|
|e53e557d5a159f5aa…|      39516|3329353.9499996677|    84.25|
11.32|[6973a06f484aacf4…|
|2b4609f8948be1887…|      36179|3171618.7699996904|    87.66|
4.22|[cc419e0650a3c5ba…|
|35afc973633aaeb6b…|      31206| 2735668.999999728|    87.66|
3.32|[d20b021d3efdf267…|
|e0d64dcfaa3b6db5c…|      31153| 5226407.629999666|   167.77|
30.9|[7d13fca152253586…|
|42a2c92a0979a949c…|      30486|1810926.0000002119|     59.4|
0.64|[813348c996469b40…|
|7c1bd920dbdf22470…|      29018|1739338.8199997821|    59.94|
2.77|[cc419e0650a3c5ba…|
|a62e25e09e05e6faf…|      28898|         3079869.0|   106.58|
1.5|[634964b17796e643…|
|5a848e4ab52fd5445…|      28737|3534363.6299997577|   122.99|
0.0|[c826c40d7b19f62a…|
|c4baedd846ed09b85…|      28166|2802044.6499998467|    99.48|
11.9|[a1043bafd471dff5…|
|b532349fe46b38fbc…|      27176| 993089.5700001451|    36.54|
1.92|[1025f0e2d44d7041…|
+--------------------+-----------+------------------+---------+----------------
+--------------------+
only showing top 20 rows
```

```
[ ]: # Monthly Revenue and Order Count Trend ----> HW

     order_purchase_timestamp ---> month
```

```
total_orders
total_revenue
avg_order_value
min_order_value
max_orderValues
```

[ ]:

```

```

[ ]:

```

```

[39]:

```python
# Customer Retention Analysis ( First & Last Order )

customer_retention_df = full_orders_df.groupBy('customer_id')\
.agg(
    first('order_purchase_timestamp').alias('first_order_date'),
    last('order_purchase_timestamp').alias('last_order_date'),
    count('order_id').alias('total_orders'),
    round(avg('price'),2).alias('aov')
)\
.orderBy(desc('total_orders'))
```

[40]:

```python
customer_retention_df.show()
```

```
[Stage 52:==============================>                                (1 + 1) / 2]

+--------------------+-------------------+-------------------+------------+-----
-+
|         customer_id|   first_order_date|    last_order_date|total_orders|
aov|
+--------------------+-------------------+-------------------+------------+-----
-+
|351e40989da90e704…|2017-07-13 10:42:37|2017-07-13 10:42:37|       11427|
85.99|
|50920f8cd0681fd86…|2018-01-27 11:28:32|2018-01-27 11:28:32|       10752|
43.82|
|9b43e2a62de9bab3a…|2017-05-25 22:27:50|2017-05-25 22:27:50|        8556|
26.4|
|270c23a11d024a44c…|2017-08-08 20:26:31|2017-08-08 20:26:31|        8001|
36.59|
|d3e82ccec3cb5f956…|2017-03-18 14:28:34|2017-03-18 14:28:34|        6876|
969.0|
|5c87184371002d49e…|2018-01-05 19:15:37|2018-01-05 19:15:37|        6876|
12.49|
|d5f2b3f597c7ccafb…|2017-12-13 14:21:15|2017-12-13 14:21:15|        6706|
59.0|
|c2f18647725395af4…|2018-03-06 19:21:47|2018-03-06 19:21:47|        6612|
34.9|
|24e7dc2ff8c071263…|2017-11-24 16:16:45|2017-11-24 16:16:45|        6597|
59.2|
|7bb57d182bdc11653…|2018-04-02 17:11:30|2018-04-02 17:11:30|        6258|
86.9|
|d22f25a9fadfb1abb…|2018-05-12 12:28:58|2018-05-12 12:28:58|        6072|
14.99|
|63b964e79dee32a35…|2018-02-14 16:34:27|2018-02-14 16:34:27|        6072|
412.0|
|1ff773612ab8934db…|2018-04-19 13:54:06|2018-04-19 13:54:06|
5820|284.99|
|13aa59158da63ba0e…|2017-09-23 14:56:45|2017-09-23 14:56:45|        5206|
79.99|
|78fc46047c4a639e8…|2017-11-28 22:24:18|2017-11-28 22:24:18|
5200|109.97|
|dd3f1762eb601f41c…|2018-07-18 12:59:21|2018-07-18 12:59:21|
4992|179.99|
|a193aa8d905b8e246…|2018-02-12 18:04:28|2018-02-12 18:04:28|        4896|
9.99|
|9eb3d566e87289dcb…|2018-06-08 16:42:11|2018-06-08 16:42:11|        4872|
5.11|
|2ba91e12e5e4c9f56…|2017-11-25 13:54:39|2017-11-25 13:54:39|        4752|
99.9|
|1b2ab6eda1946a6ff…|2017-11-24 10:41:43|2017-11-24 10:41:43|        4728|
32.99|
+--------------------+-------------------+-------------------+------------+-----
-+
only showing top 20 rows
```

[ ]:
```
# HW - Correct the last_order_date
```

[ ]:

[ ]:

[ ]:

5 Extended Enrichment

[ ]:

[43]:
```
# Order Status Flags

full_orders_df = full_orders_df.withColumn('is_delivered',␣
↪when(col('order_status')== 'delivered',lit(1)).otherwise(lit(0)))\
.withColumn('is_canceled', when(col('order_status')== 'canceled',lit(1)).
 ↪otherwise(lit(0)))
```

[46]:
```
full_orders_df.where(full_orders_df['order_status']=='canceled').
 ↪select('order_status','is_delivered','is_canceled').show(100)
```

```
+------------+------------+-----------+
|order_status|is_delivered|is_canceled|
+------------+------------+-----------+
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|           1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
|    canceled|           0|          1|
+------------+------------+-----------+
only showing top 100 rows
```

[47]:
```python
# Order Revenue Calcualtion

full_orders_df = full_orders_df.
 ↪withColumn('order_revenue',col('price')+col('freight_value'))
```

[48]:
```python
full_orders_df.select('price','freight_value','order_revenue').show()
```

```
+-----+-------------+-------------+
|price|freight_value|order_revenue|
+-----+-------------+-------------+
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
|29.99|         8.72|        38.71|
+-----+-------------+-------------+
only showing top 20 rows
```

[52]:
```
customer_spending_df.printSchema()
```

```
root
 |-- customer_id: string (nullable = true)
 |-- total_orders: long (nullable = false)
 |-- total_spent: double (nullable = true)
 |-- AOV: double (nullable = true)
```

[57]:
```
# Customer Segmentation based on spending

customer_spending_df = customer_spending_df.withColumn(
    'customer_segment',
    when(col('AOV') >=1200,"High-Value")
    .when( (col('AOV')<1200) & (col('AOV') >=700),'Medium_Value')
    .otherwise('Low-Value'))
```

[58]:
```
customer_spending_df.show()
```

```
[Stage 66:==============================>                              (1 + 1) / 2]

+--------------------+------------+------------------+-------+----------------+
|         customer_id|total_orders|       total_spent|    AOV|customer_segment|
+--------------------+------------+------------------+-------+----------------+
|d3e82ccec3cb5f956...|        6876|         6662844.0|  969.0|    Medium_Value|
|df55c14d1476a9a34...|         743|         3565657.0| 4799.0|      High-Value|
|fe5113a38e3575c04...|        2292|         3293604.0| 1437.0|      High-Value|
|ec5b2ba62e5743423...|        1428|         2556120.0| 1790.0|      High-Value|
|63b964e79dee32a35...|        6072|         2501664.0|  412.0|       Low-Value|
|46bb3c0b1a65c8399...|         748|         2336752.0| 3124.0|      High-Value|
|05455dfa7cd02f13d...|        2184| 2160194.400000087|  989.1|    Medium_Value|
|3690e975641f01bd0...|         802|         2124498.0| 2649.0|      High-Value|
|349509b216bd5ec11...|         743|         1923627.0| 2589.0|      High-Value|
|695476b5848d64ba0...|         687|1820543.1299999943|2649.99|      High-Value|
|73236a0796f53d60d...|         832|         1755520.0| 2110.0|      High-Value|
|cc803a2c412833101...|         762|         1676400.0| 2200.0|      High-Value|
|1ff773612ab8934db...|        5820|1658641.7999999512| 284.99|       Low-Value|
|fced842c7dad61e8c...|         602|         1654898.0| 2749.0|      High-Value|
|1ecb47d23dc8203cd...|        1164|1629588.3599999903|1399.99|      High-Value|
|de832e8dbb1f588a4...|        2190|1584990.5999999817| 723.74|    Medium_Value|
|803cd9b04f9cd252c...|         488|         1512312.0| 3099.0|      High-Value|
|d72181923840c8895...|        2721|1488114.8999999566|  546.9|       Low-Value|
|06d478ba352a27a51...|        1146|         1461150.0| 1275.0|      High-Value|
|0049e8442c2a3e4a8...|        1204|         1444800.0| 1200.0|      High-Value|
+--------------------+------------+------------------+-------+----------------+
only showing top 20 rows
```

```
[59]: full_orders_df = full_orders_df.join(customer_spending_df.
        ↪select('customer_id','customer_segment'),'customer_id',how='left')
```

```
[62]: full_orders_df.select('customer_id','customer_segment').show()
```

```
+--------------------+----------------+
|         customer_id|customer_segment|
+--------------------+----------------+
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
|9ef432eb625129730…|       Low-Value|
+--------------------+----------------+
only showing top 20 rows
```

```
[63]: full_orders_df.select('order_purchase_timestamp').show()
```

```
+------------------------+
|order_purchase_timestamp|
+------------------------+
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
|     2017-10-02 10:56:33|
+------------------------+
only showing top 20 rows
```

[65]:
```
#Hourly Order Distribution
full_orders_df = full_orders_df.
  ↪withColumn('hour_of_day',expr('hour(order_purchase_timestamp)'))
```

[67]:
```
full_orders_df.select('order_purchase_timestamp','hour_of_day').show()
```

```
+------------------------+-----------+
|order_purchase_timestamp|hour_of_day|
+------------------------+-----------+
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
|     2017-10-02 10:56:33|         10|
+------------------------+-----------+
only showing top 20 rows
```

[72]:
```
# Weekday vs Weekend Order

full_orders_df = full_orders_df.withColumn('order_day_type',\
                                                               ␣
 ↪when(dayofweek('order_purchase_timestamp').isin(1,7),lit('Weekend')).
 ↪otherwise(lit('weekday')))
```

[73]:
```
full_orders_df.select('order_purchase_timestamp','order_day_type').show()
```

```
+------------------------+--------------+
|order_purchase_timestamp|order_day_type|
+------------------------+--------------+
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
|     2017-10-02 10:56:33|       weekday|
+------------------------+--------------+
only showing top 20 rows
```

[74]:
```
full_orders_df.printSchema()
```

```
root
 |-- customer_id: string (nullable = true)
 |-- order_id: string (nullable = true)
 |-- seller_id: string (nullable = true)
 |-- product_id: string (nullable = true)
 |-- order_status: string (nullable = true)
 |-- order_purchase_timestamp: timestamp (nullable = true)
 |-- order_approved_at: timestamp (nullable = true)
 |-- order_delivered_carrier_date: timestamp (nullable = true)
 |-- order_delivered_customer_date: timestamp (nullable = true)
 |-- order_estimated_delivery_date: timestamp (nullable = true)
 |-- order_item_id: integer (nullable = true)
 |-- shipping_limit_date: timestamp (nullable = true)
 |-- price: double (nullable = true)
 |-- freight_value: double (nullable = true)
 |-- product_category_name: string (nullable = true)
 |-- product_name_lenght: integer (nullable = true)
 |-- product_description_lenght: integer (nullable = true)
 |-- product_photos_qty: integer (nullable = true)
 |-- product_weight_g: integer (nullable = true)
 |-- product_length_cm: integer (nullable = true)
 |-- product_height_cm: integer (nullable = true)
 |-- product_width_cm: integer (nullable = true)
 |-- seller_zip_code_prefix: integer (nullable = true)
 |-- seller_city: string (nullable = true)
 |-- seller_state: string (nullable = true)
 |-- customer_unique_id: string (nullable = true)
 |-- customer_zip_code_prefix: integer (nullable = true)
 |-- customer_city: string (nullable = true)
 |-- customer_state: string (nullable = true)
 |-- geolocation_zip_code_prefix: integer (nullable = true)
 |-- geolocation_lat: double (nullable = true)
 |-- geolocation_lng: double (nullable = true)
 |-- geolocation_city: string (nullable = true)
 |-- geolocation_state: string (nullable = true)
 |-- review_id: string (nullable = true)
 |-- review_score: string (nullable = true)
 |-- review_comment_title: string (nullable = true)
 |-- review_comment_message: string (nullable = true)
 |-- review_creation_date: string (nullable = true)
 |-- review_answer_timestamp: string (nullable = true)
 |-- payment_sequential: integer (nullable = true)
 |-- payment_type: string (nullable = true)
 |-- payment_installments: integer (nullable = true)
 |-- payment_value: double (nullable = true)
 |-- is_delivered: integer (nullable = false)
 |-- is_canceled: integer (nullable = false)
 |-- order_revenue: double (nullable = true)
 |-- customer_segment: string (nullable = true)
 |-- hour_of_day: integer (nullable = true)
 |-- order_day_type: string (nullable = false)
```

[ ]:
```
# a new column frieght category  based on freight_value --> low, med or high
```

[ ]:
```
# Order Volume by Customer State
```

[ ]:
```

```

[75]:
```
!hadoop fs -mkdir /olist/processed/
```

[79]:
```
full_orders_df.write.mode('overwrite').parquet('/olist/processed')
```

[78]:
```
!hadoop fs -ls -h /olist/processed/
```

```
Found 11 items
-rw-r--r--   2 root hadoop          0 2025-02-28 14:57 /olist/processed/_SUCCESS
-rw-r--r--   2 root hadoop     30.8 M 2025-02-28 14:57 /olist/processed/part-
00000-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     31.1 M 2025-02-28 14:57 /olist/processed/part-
00001-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     30.3 M 2025-02-28 14:57 /olist/processed/part-
00002-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     31.1 M 2025-02-28 14:57 /olist/processed/part-
00003-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     31.1 M 2025-02-28 14:57 /olist/processed/part-
00004-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     18.6 M 2025-02-28 14:57 /olist/processed/part-
00005-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     19.2 M 2025-02-28 14:57 /olist/processed/part-
00006-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     18.9 M 2025-02-28 14:57 /olist/processed/part-
00007-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     18.6 M 2025-02-28 14:57 /olist/processed/part-
00008-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
-rw-r--r--   2 root hadoop     18.8 M 2025-02-28 14:57 /olist/processed/part-
00009-5303d6eb-c739-4c66-9f5f-d566a672389c-c000.snappy.parquet
```