- 1.1 Download the relevant dataset from the "**Course Resources**" section or the project description
- 1.2 Upload the dataset to the **"FTP"** lab from your local system
- 1.3 To move the dataset to **"HDFS"** from the "Webconsole" use the put command

```
[therohitsaha08gmail@sl-cdp-prod-en21 ~]$ hdfs dfs -ls Market_Analysis/insta-cart
Found 7 items
-rw-rw-r--   2 therohitsaha08gmail hadoop       2603 2024-03-02 10:10 Market_Analysis/insta-cart/aisles.csv
-rw-rw-r--   2 therohitsaha08gmail hadoop        270 2024-03-02 10:10 Market_Analysis/insta-cart/departments.csv
-rw-rw-r--   2 therohitsaha08gmail hadoop  577550706 2024-03-02 10:10 Market_Analysis/insta-cart/order_products__prior.csv
-rw-rw-r--   2 therohitsaha08gmail hadoop   24680147 2024-03-02 10:10 Market_Analysis/insta-cart/order_products__train.csv
-rw-rw-r--   2 therohitsaha08gmail hadoop  108968645 2024-03-02 10:10 Market_Analysis/insta-cart/orders.csv
-rw-rw-r--   2 therohitsaha08gmail hadoop    2166953 2024-03-02 10:10 Market_Analysis/insta-cart/products.csv
-rw-rw-r--   2 therohitsaha08gmail hadoop    1475693 2024-03-02 10:10 Market_Analysis/insta-cart/sample_submission.csv
```

- Login to the Pyspark shell
- Explore the orders CSV file and create a DataFrame
    - Read the orders data as a DataFrame in PySpark

  Note: The column "**days_since_prior_order**" may contain NULL values

- Display the data up to 10 rows
- Replace all null values with a dummy "999" value in the DataFrame that was created in task 1
- Examine the orders CSV file and find the busiest day of the week by reading the data as a PySpark DataFrame
- Display the result that contains the total orders placed on each day of the week (Monday to Sunday)

```
>>> df_order = spark.read.format("csv").option("delimiter", ",").option("header", "true").option("inferSchema", "true").load("Market_Analysis/insta-cart/orders.csv")
>>> df_order.show()
+--------+-------+--------+------------+---------+-----------------+---------------------+
|order_id|user_id|eval_set|order_number|order_dow|order_hour_of_day|days_since_prior_order|
+--------+-------+--------+------------+---------+-----------------+---------------------+
| 2539329|      1|   prior|           1|        2|                8|                 null|
| 2398795|      1|   prior|           2|        3|                7|                 15.0|
|  473747|      1|   prior|           3|        3|               12|                 21.0|
| 2254736|      1|   prior|           4|        4|                7|                 29.0|
|  431534|      1|   prior|           5|        4|               15|                 28.0|
| 3367565|      1|   prior|           6|        2|                7|                 19.0|
|  550135|      1|   prior|           7|        1|                9|                 20.0|
| 3108588|      1|   prior|           8|        1|               14|                 14.0|
| 2295261|      1|   prior|           9|        1|               16|                  0.0|
| 2550362|      1|   prior|          10|        4|                8|                 30.0|
| 1187899|      1|   train|          11|        4|                8|                 14.0|
| 2168274|      2|   prior|           1|        2|               11|                 null|
| 1501582|      2|   prior|           2|        5|               10|                 10.0|
| 1901567|      2|   prior|           3|        1|               10|                  3.0|
|  738281|      2|   prior|           4|        2|               10|                  8.0|
| 1673511|      2|   prior|           5|        3|               11|                  8.0|
| 1199898|      2|   prior|           6|        2|                9|                 13.0|
| 3194192|      2|   prior|           7|        2|               12|                 14.0|
|  788338|      2|   prior|           8|        1|               15|                 27.0|
| 1718559|      2|   prior|           9|        2|                9|                  8.0|
+--------+-------+--------+------------+---------+-----------------+---------------------+
only showing top 20 rows

>>> orderNew_df = df_order.withColumn("days_since_prior_order", when(col("days_since_prior_order").isNull(),"999").otherwise(col("days_since_prior_order"))).show()
+--------+-------+--------+------------+---------+-----------------+---------------------+
|order_id|user_id|eval_set|order_number|order_dow|order_hour_of_day|days_since_prior_order|
+--------+-------+--------+------------+---------+-----------------+---------------------+
| 2539329|      1|   prior|           1|        2|                8|                  999|
| 2398795|      1|   prior|           2|        3|                7|                 15.0|
|  473747|      1|   prior|           3|        3|               12|                 21.0|
| 2254736|      1|   prior|           4|        4|                7|                 29.0|
|  431534|      1|   prior|           5|        4|               15|                 28.0|
| 3367565|      1|   prior|           6|        2|                7|                 19.0|
|  550135|      1|   prior|           7|        1|                9|                 20.0|
| 3108588|      1|   prior|           8|        1|               14|                 14.0|
| 2295261|      1|   prior|           9|        1|               16|                  0.0|
| 2550362|      1|   prior|          10|        4|                8|                 30.0|
| 1187899|      1|   train|          11|        4|                8|                 14.0|
| 2168274|      2|   prior|           1|        2|               11|                  999|
| 1501582|      2|   prior|           2|        5|               10|                 10.0|
| 1901567|      2|   prior|           3|        1|               10|                  3.0|
|  738281|      2|   prior|           4|        2|               10|                  8.0|
| 1673511|      2|   prior|           5|        3|               11|                  8.0|
| 1199898|      2|   prior|           6|        2|                9|                 13.0|
```

```
>>> day_df = df_order.groupBy(col("order_dow").alias("Day_of_the_Week")).count().orderBy("order_dow").select(col("Day_of_the_Week"),col("count").alias("total_orders"))
>>> day_df.show()
+--------------+------------+
|Day_of_the_Week|total_orders|
+--------------+------------+
|             0|      600905|
|             1|      587478|
|             2|      467260|
|             3|      436972|
|             4|      426339|
|             5|      453368|
|             6|      448761|
+--------------+------------+

>>> DayofWeek_df = day_df.withColumn("Day of the Week", expr("CASE WHEN Day_of_the_Week = 0 THEN 'Sunday' " + "WHEN Day_of_the_Week = 1 THEN 'Monday' " + "WHEN Day_of_the_Week = 2 THEN 'Tuesday' " + "WHEN Day_of_the_Week = 3 THEN
'Wednesday' " + "WHEN Day_of_the_Week = 4 THEN 'Thursday' " + "WHEN Day_of_the_Week = 5 THEN 'Friday' " + "WHEN Day_of_the_Week = 6 THEN 'Saturday' END")).select(col("Day of the Week"),col("total_orders"))
>>> DayofWeek_df.show()
+--------------+------------+
|Day of the Week|total_orders|
+--------------+------------+
|        Sunday|      600905|
|        Monday|      587478|
|       Tuesday|      467260|
|     Wednesday|      436972|
|      Thursday|      426339|
|        Friday|      453368|
|      Saturday|      448761|
+--------------+------------+
```

- Give a breakdown of orders by the hour and identify the busiest hour
    - Select the number of order IDs as "**Total_Orders**" and the hour at which the order was placed
    - Display the result that contains total orders and the hour

```
>>> Hour_df = df_order.groupBy(col("order_hour_of_day").alias("Hour")).count().orderBy(col("Hour")).select(col("Hour"),col("count").alias("Total_Orders"))
>>> Hour_df.show()
+----+------------+
|Hour|Total_Orders|
+----+------------+
|   0|       22758|
|   1|       12398|
|   2|        7539|
|   3|        5474|
|   4|        5527|
|   5|        9569|
|   6|       30529|
|   7|       91868|
|   8|      178201|
|   9|      257812|
|  10|      288418|
|  11|      284728|
|  12|      272841|
|  13|      277999|
|  14|      283042|
|  15|      283639|
|  16|      272553|
|  17|      228795|
|  18|      182912|
|  19|      140569|
+----+------------+
only showing top 20 rows

>>> BusiestHour_df = Hour_df.orderBy(col("Total_Orders").desc()).limit(1)
>>> BusiestHour_df.show()
+----+------------+
|Hour|Total_Orders|
+----+------------+
|  10|      288418|
+----+------------+
```

- Identify the most popular item based on the order count by exploring order_products__prior and products datasets.
- Calculate the top 10 popular items based on the count of orders.
- Display the result that contains the product name as "Popular_product_name" and the count of order id as "Order_Count".

```
>>> products_prior_df = spark.read.format("csv").option("delimiter", ",").option("header", "true").option("inferSchema", "true").load("Market_Analysis/insta-cart/order_products__prior.csv")
>>> products_df = spark.read.format("csv").option("delimiter", ",").option("header", "true").option("inferSchema", "true").load("Market_Analysis/insta-cart/products.csv")
>>> popular_df = products_prior_df.join(products_df, "product_id").groupBy(col("product_name").alias("popular_product_name")).agg(count(col("order_id")).alias("order_count")).orderBy(col("order_count").desc()).limit(10)
>>> popular_df.show()
+--------------------+-----------+
|popular_product_name|order_count|
+--------------------+-----------+
|              Banana|     472565|
|Bag of Organic Ba...|     379450|
|Organic Strawberries|     264683|
|Organic Baby Spinach|     241921|
|Organic Hass Avocado|     213584|
|     Organic Avocado|     176815|
|         Large Lemon|     152657|
|       Strawberries|     142951|
|               Limes|     140627|
|  Organic Whole Milk|     137905|
+--------------------+-----------+
```

- Explore the department dataset and create a DataFrame.
- Recognize the department which has published the maximum products.
- Display the department ID that has published the maximum products.

```
>>> department_df = spark.read.format("csv").option("delimiter", ",").option("header", "true").option("inferSchema", "true").load("Market_Analysis/insta-cart/departments.csv")
>>> max_prod_dept_df = products_df.join(department_df, "department_id").groupBy(col("department_id"), col("department")).agg(count(col("product_name")).alias("max_products")).orderBy(col("max_products").desc())
>>> max_prod_dept_df.show()
+-------------+--------------+------------+
|department_id|    department|max_products|
+-------------+--------------+------------+
|           11| personal care|        6563|
|           19|        snacks|        6264|
|           13|       pantry|        5371|
|            7|     beverages|        4365|
|            1|        frozen|        4007|
|           16|    dairy eggs|        3449|
|           17|     household|        3084|
|           15|  canned goods|        2092|
|            9|dry goods pasta|        1858|
|            4|       produce|        1684|
|            3|        bakery|        1516|
|           20|          deli|        1322|
|           21|       missing|        1258|
|            6| international|        1139|
|           14|     breakfast|        1115|
|           18|        babies|        1081|
|            5|       alcohol|        1054|
|            8|          pets|         972|
|           12|  meat seafood|         907|
|            2|         other|         548|
+-------------+--------------+------------+
only showing top 20 rows
```