Question 1:

1. Try word count program using dataframe

```
object wordCountdf extends App {
  val sparkconf = new SparkConf()
  sparkconf set("spark.app","wordCountDF")
  sparkconf.set("spark.master","local[1]")

  val spark = SparkSession.builder().config(sparkconf).getOrCreate()

  val wordCountSchemaDDL = "id Int, descriptions String"

  val wordCountDataFrame = spark.read.option("header",
true).schema(wordCountSchemaDDL).csv("C:\\Users\\solom\\Desktop\\solomonITC\\Demo\\input\
\wordCountTask.csv")
  //wordCountDataFrame.show()

  var withcol = wordCountDataFrame.withColumn(colName = "wordCount",
functions.size(functions.split(functions.col("descriptions"), " "))-1)
  withcol.show()
//  withcol.repartition(numPartitions =
1).write.csv("C:\\Users\\solom\\Desktop\\solomonITC\\Demo\\output\\dataFrameWordCount1")
}
```

OutPut:

```
+---+--------------------+---------+
| id|       descriptions|wordCount|
+---+--------------------+---------+
|  1|         I love you|        3|
|  2|     I love you too|        4|
|  3| I love where I a...|        8|
```

```
+---+--------------------+---------+
```

Quesytion 2:

-- 2. What is the total amount each customer spent at the restaurant?

```
val salesTableSchemaDDL = "  customer_id String, order_date Date,product_id Int"
val menuTableSchemaDDL = "product_id Int, product_name String,price Int"
val membersTableSchemaDDL = "customer_id String, join_date Date"

val salesDataFrame = spark.read.option("header",
true).schema(salesTableSchemaDDL).csv("C:\\Users\\solom\\Desktop\\solomonITC\\Demo\\input
\\sales.csv")
//salesDataFrame.show()

val menuDataFrame = spark.read.option("header",
true).schema(menuTableSchemaDDL).csv("C:\\Users\\solom\\Desktop\\solomonITC\\Demo\\input\
\menu.csv")
//menuDataFrame.show()

val membersDataFrame = spark.read.option("header",
true).schema(membersTableSchemaDDL).csv("C:\\Users\\solom\\Desktop\\solomonITC\\Demo\\inp
ut\\members.csv")
//membersDataFrame.show()

//What is the total amount each customer spent at the restaurant?



val salesMenuTableDF = salesDataFrame.join(menuDataFrame, salesDataFrame("product_id")
=== menuDataFrame("product_id"))

val totalBillAmount =
salesMenuTableDF.groupBy("customer_id").agg(sum(col("price")).alias("total_bill_amount"))
totalBillAmount.show()
```

OutPut:

```
+-----------+----------------+
```

|customer_id|total_bill_amount|

+-----------+-----------------+

| B| 54|

| C| 36|

| A| 66|

+-----------+-----------------+

-- 3. How many days has each customer visited the restaurant?

Code :

```
//How many days has each customer visited the restaurant?
val dayVisitsEachCustomer = salesDataFrame.select("customer_id",
"order_date").distinct().groupBy("customer_id")
  .agg(count("order_date").alias("total_day_visits"))
dayVisitsEachCustomer.show()
```

OutPut :

+-----------+---------------+

|customer_id|total_day_visits|

+-----------+---------------+

| B| 6|

| C| 2|

| A| 4|

+-----------+---------------+

-- 4. What was the first item from the menu purchased by each customer?

Code :

```
//What was the first item from the menu purchased by each customer?

val firstItemPurchaseDateDF =
salesMenuTableDF.groupBy("customer_id").agg(min(col("order_date")).alias("purchasedate"))

val firstItemCutomerPurDateDF = salesMenuTableDF.join(firstItemPurchaseDateDF,
Seq("customer_id"), "inner")
  .filter(col("order_date") === col("purchasedate"))
  .select("customer_id", "product_name")

firstItemCutomerPurDateDF.show()
```

OutPut :

```
+-----------+------------+
|customer_id|product_name|
+-----------+------------+
|          A|      curry|
|          B|      sushi|
|          C|       ramen|
+-----------+------------+
```

-- 4. What is the most purchased item on the menu and how many times was it purchased by all customers?

Code :

```
val purchasedItemDF =
salesMenuTableDF.groupBy("product_name").agg(count("*").alias("purchases_item"))
  .orderBy(desc("purchases_item")).limit(1)

  purchasedItemDF.show()
```

OutPut :

```
+------------+--------------+
|product_name|purchases_item|
+------------+--------------+
|     ramen|         8|
+------------+--------------+
```

-- 5. Which item was the most popular for each customer?

Code :

```
val dfForMostPopularItem =
salesMenuTableDF.groupBy("customer_id","product_name").agg(count("*").alias("totalProduct
"))
  .withColumn("rankOrder",
rank().over(Window.partitionBy("customer_id").orderBy(desc("totalProduct"))))
  .filter(col("rankOrder") === 1 )
dfForMostPopularItem.show()
```

OutPut :

```
+-----------+------------+------------+---------+
|customer_id|product_name|totalProduct|rankOrder|
+-----------+------------+------------+---------+
|          B|      ramen|           2|        1|
|          B|      curry|           2|        1|
|          C|      ramen|           3|        1|
```

| A| ramen| 3| 1|

+-----------+------------+------------+---------+

-- 6. Which item was purchased first by the customer after they became a member?

Code:

```
val salesMembersTableDF = salesDataFrame.join(membersDataFrame, ("customer_id"))

val memberFirstPurchaseDF = salesMembersTableDF.filter(col("order_date") >=
col("join_date"))
  .withColumn("rank_order",
    rank().over(Window.partitionBy("customer_id").orderBy("order_date")))
  .filter(col("rank_order") === 1)
  .groupBy("customer_id")
  .agg(first("product_name").alias("memberFirstPurchase"))

memberFirstPurchaseDF.show()
```

+-----------+--------------------+

|customer_id|memberFirstPurchase|

+-----------+--------------------+

| A| curry|

+-----------+--------------------+

-- 7. Which item was purchased just before the customer became a member?

-- 8. What is the total items and amount spent for each member before they became a member?

-- 9. If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?

-- 10. In the first week after a customer joins the program (including their join date) they earn 2x points on all items, not just sushi - how many points do customer A and B have at the end of January?