Q1]

Input File :-

```
scala> root@5a8ed1a45fb6:/# cat input7.txt
Name ~|Age
Amira,Sheik~|30
Rosy,Clarke~|27
Kaira,Rao~|28
Andrew,Simod~|37
George,Bush~|60
Fintoff,david~|12
Adam,James~|22
root@5a8ed1a45fb6:/#
```

Code :-

```
import org.apache.spark.sql.{SparkSession, types}
import org.apache.spark.sql.types.{StructType, StructField, StringType, Integer>

// Create a SparkSession
val spark = SparkSession.builder.appName("LoadData").getOrCreate()

// Define the schema
val schema = StructType(Array(
   StructField("Name", StringType, nullable = true),
   StructField("Age", IntegerType, nullable = true)
))

// Load the data
val df = spark.read.format("csv").option("delimiter", "~|").schema(schema).load>


// Define the schema
val schema = StructType(Array(
   StructField("Name", StringType, nullable = true),
   StructField("Age", IntegerType, nullable = true)
))

// Load the data
val df = spark.read.format("csv").option("delimiter", "~|").schema(schema).load>

// Display the schema
df.printSchema()

// Show the first 5 rows
df.show(5)

// Filter and show people of age greater than 30
```

```
df.filter(df("Age") > 30).show()
```

Output :-

```
scala> :load Q8.scala
Loading Q8.scala...
import org.apache.spark.sql.{SparkSession, types}
import org.apache.spark.sql.types.{StructType, StructField, StringType, IntegerType}
24/04/15 10:17:36 WARN SparkSession: Using an existing Spark session; only runtime SQL
configurations will take effect.
spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@34a7decd
schema: org.apache.spark.sql.types.StructType =
StructType(StructField(Name,StringType,true),StructField(Age,IntegerType,true))
df: org.apache.spark.sql.DataFrame = [Name: string, Age: int]
root
 |-- Name: string (nullable = true)
 |-- Age: integer (nullable = true)

+------------+----+
|        Name| Age|
+------------+----+
|       Name |null|
| Amira,Sheik|  30|
| Rosy,Clarke|  27|
|   Kaira,Rao|  28|
|Andrew,Simod|  37|
+------------+----+
only showing top 5 rows

+------------+---+
|        Name|Age|
+------------+---+
|Andrew,Simod| 37|
| George,Bush| 60|
```

2]

I]

Input File :-

```
Model1,Company1,Black,Over-Ear,4.5,1200,1500,2000,500
Model2,Company2,White,In-Ear,4.0,800,1800,2500,700
Model3,Company3,Blue,On-Ear,4.2,500,1300,2100,800
Model4,Company4,Red,Over-Ear,3.7,1500,2000,2800,800
```

```
Model5,Company5,Black,In-Ear,3.3,900,1600,2400,800
```

Code :-

```
-- Load the data from the file
data = LOAD 'input.txt' USING PigStorage(',')
   AS (Model:chararray, Company:chararray, Color:chararray, Type:chararray, Ra>

-- Filter the data where discount is less than 800 and color is either black or>
filtered_data = FILTER data BY (Discount < 800) AND (Color == 'Black' OR Color >

-- Store the result in an output file
STORE filtered_data INTO 'pigout';
```

Output

```
root@5a8ed1a45fb6:/pigout# cat part-m-00000
Model1          Company1        Black   Over-Ear        4.5     1200    1500    2000    500
Model2          Company2        White   In-Ear  4.0     800     1800    2500    700
```

II]
Code :-

```
  GNU nano 6.2                                                        pigscript4.pig
-- Load the data from the file
data = LOAD 'input.txt' USING PigStorage(',')
   AS (Model:chararray, Company:chararray, Color:chararray, Type:chararray, Rating:float,
Number_of_Ratings:int, Sell_Price:int, MRP:int, Discount:int);

-- Filter the data where rating is greater than 4
filtered_data = FILTER data BY Rating > 4;

-- Project the company names
company_names = FOREACH filtered_data GENERATE Company;

-- Store the result in an output file
STORE company_names INTO 'outpig2';
```

output

```
root@5a8ed1a45fb6:/outpig2# cat part-m-00000
Company1
Company3
```