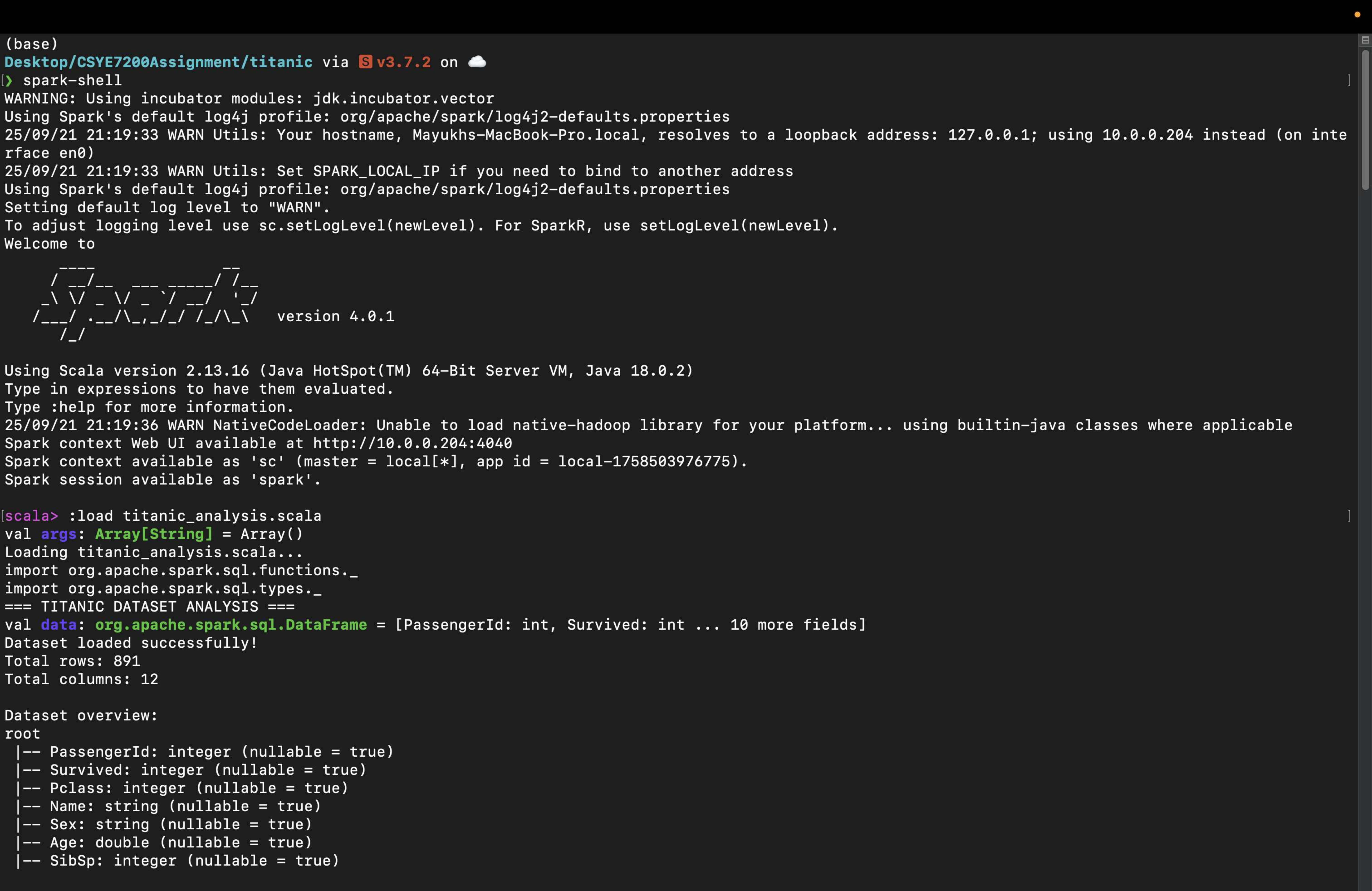
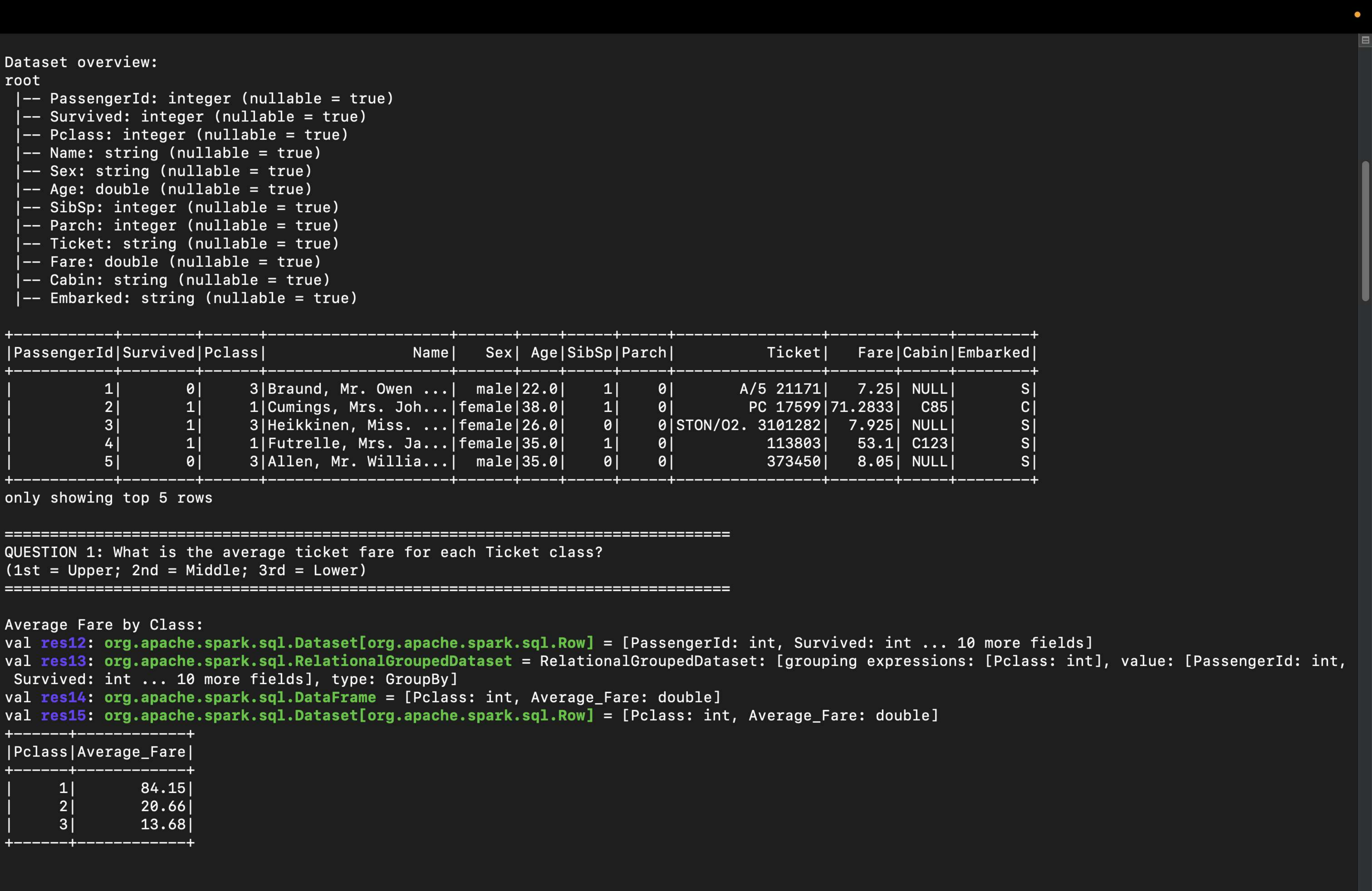
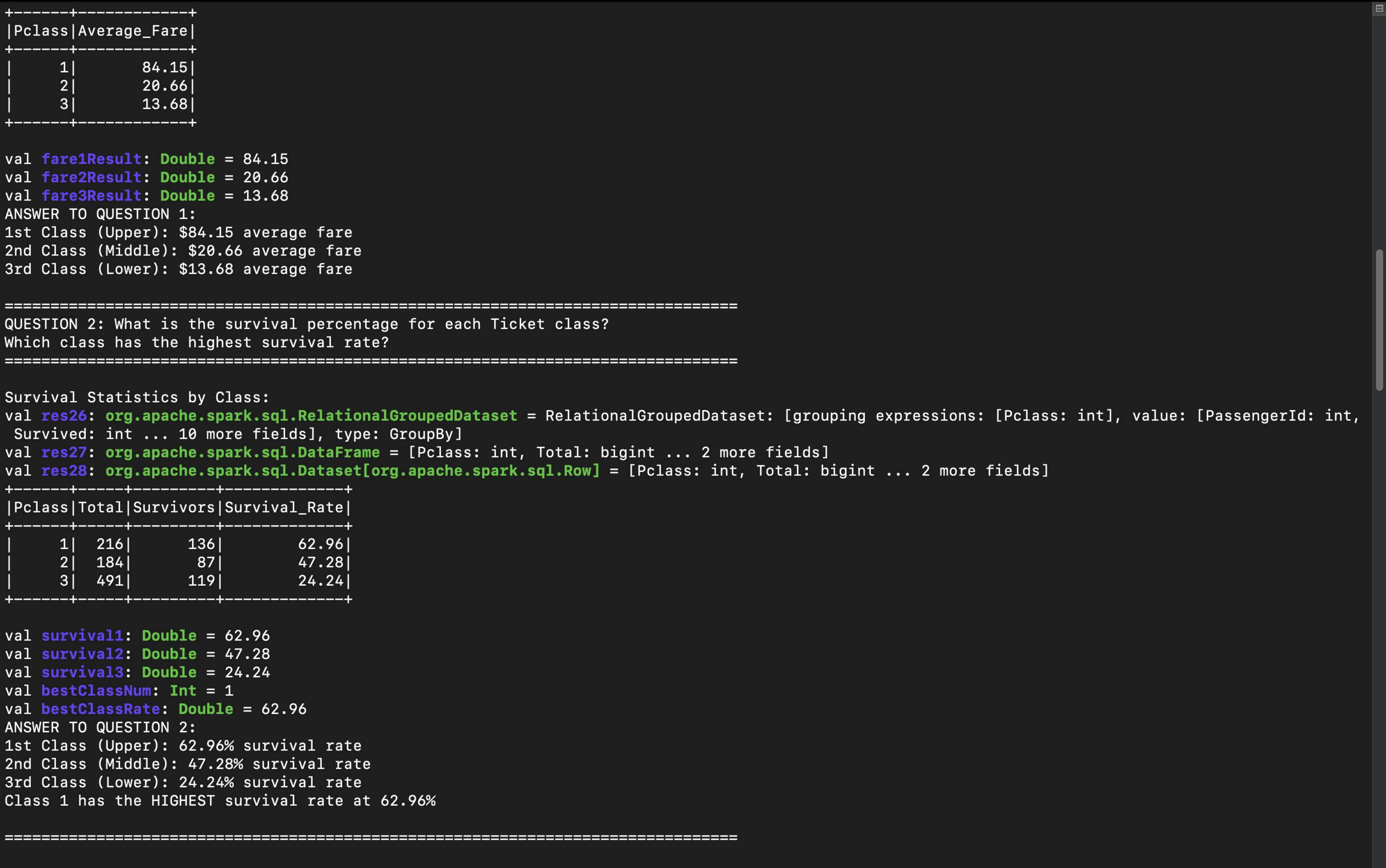


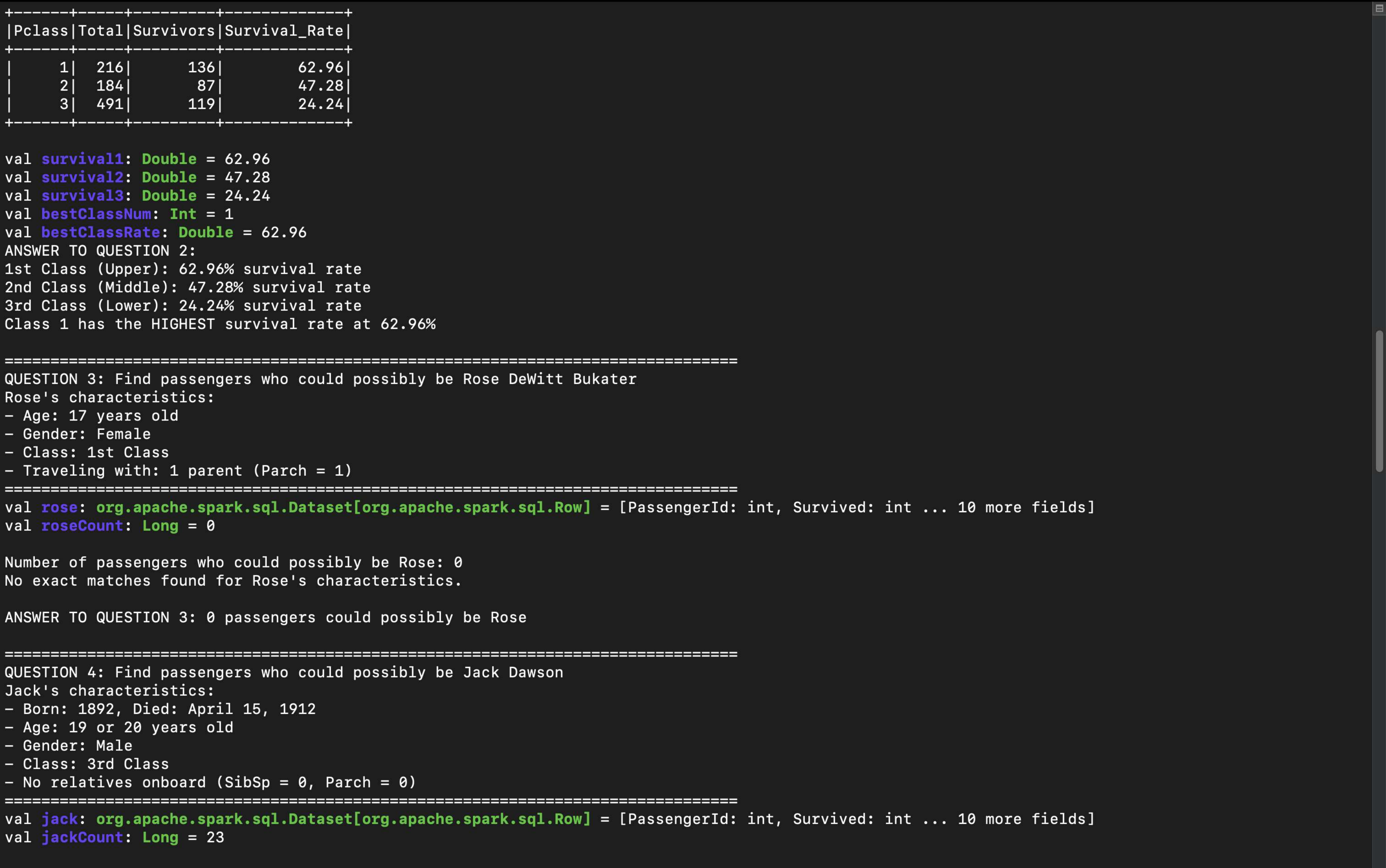
```
(base)
Desktop/CSYE7200Assignment/titanic via S v3.7.2 on ☁
❯ spark-shell
WARNING: Using incubator modules: jdk.incubator.vector
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
25/09/21 21:19:33 WARN Utils: Your hostname, Mayukhs-MacBook-Pro.local, resolves to a loopback address: 127.0.0.1; using 10.0.0.204 instead (on inte
rface en0)
25/09/21 21:19:33 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 4.0.1
      /_/

Using Scala version 2.13.16 (Java HotSpot(TM) 64-Bit Server VM, Java 18.0.2)
Type in expressions to have them evaluated.
Type :help for more information.
25/09/21 21:19:36 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://10.0.0.204:4040
Spark context available as 'sc' (master = local[*], app id = local-1758503976775).
Spark session available as 'spark'.

scala> :load titanic_analysis.scala
val args: Array[String] = Array()
Loading titanic_analysis.scala...
import org.apache.spark.sql.functions._
import org.apache.spark.sql.types._
=== TITANIC DATASET ANALYSIS ===
val data: org.apache.spark.sql.DataFrame = [PassengerId: int, Survived: int ... 10 more fields]
Dataset loaded successfully!
Total rows: 891
Total columns: 12

Dataset overview:
root
 |-- PassengerId: integer (nullable = true)
 |-- Survived: integer (nullable = true)
 |-- Pclass: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- SibSp: integer (nullable = true)
```



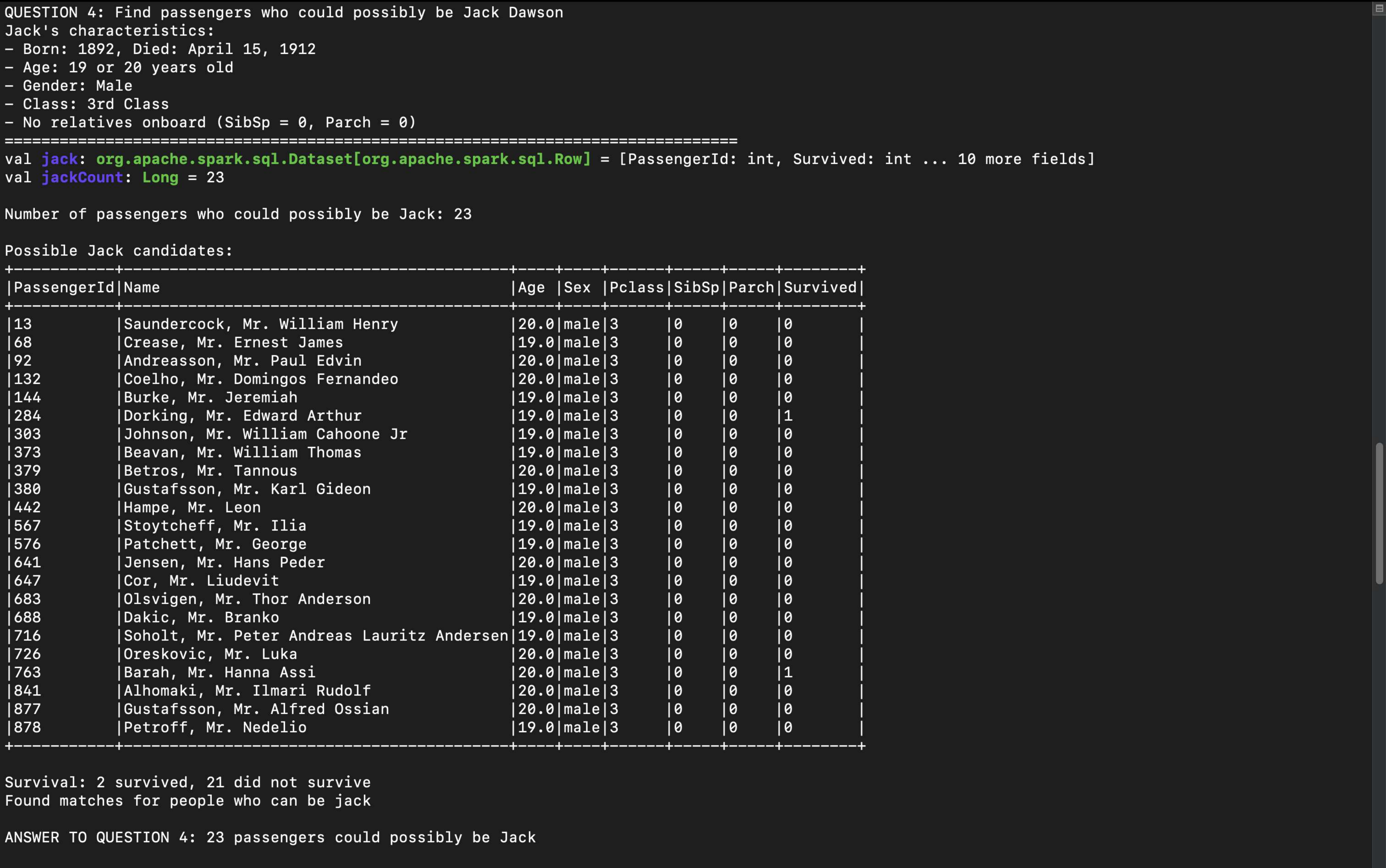
```
Dataset overview:
root
 |-- PassengerId: integer (nullable = true)
 |-- Survived: integer (nullable = true)
 |-- Pclass: integer (nullable = true)
 |-- Name: string (nullable = true)
 |-- Sex: string (nullable = true)
 |-- Age: double (nullable = true)
 |-- SibSp: integer (nullable = true)
 |-- Parch: integer (nullable = true)
 |-- Ticket: string (nullable = true)
 |-- Fare: double (nullable = true)
 |-- Cabin: string (nullable = true)
 |-- Embarked: string (nullable = true)

+-----------+--------+------+--------------------+------+----+-----+-----+----------------+-------+-----+--------+
|PassengerId|Survived|Pclass|                Name|   Sex| Age|SibSp|Parch|          Ticket|   Fare|Cabin|Embarked|
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+-------+-----+--------+
|          1|       0|     3|Braund, Mr. Owen ...|  male|22.0|    1|    0|       A/5 21171|   7.25| NULL|       S|
|          2|       1|     1|Cumings, Mrs. Joh...|female|38.0|    1|    0|        PC 17599|71.2833|  C85|       C|
|          3|       1|     3|Heikkinen, Miss. ...|female|26.0|    0|    0|STON/O2. 3101282|  7.925| NULL|       S|
|          4|       1|     1|Futrelle, Mrs. Ja...|female|35.0|    1|    0|          113803|   53.1| C123|       S|
|          5|       0|     3|Allen, Mr. Willia...|  male|35.0|    0|    0|          373450|   8.05| NULL|       S|
+-----------+--------+------+--------------------+------+----+-----+-----+----------------+-------+-----+--------+
only showing top 5 rows

================================================================================
QUESTION 1: What is the average ticket fare for each Ticket class?
(1st = Upper; 2nd = Middle; 3rd = Lower)
================================================================================

Average Fare by Class:
val res12: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [PassengerId: int, Survived: int ... 10 more fields]
val res13: org.apache.spark.sql.RelationalGroupedDataset = RelationalGroupedDataset: [grouping expressions: [Pclass: int], value: [PassengerId: int,
 Survived: int ... 10 more fields], type: GroupBy]
val res14: org.apache.spark.sql.DataFrame = [Pclass: int, Average_Fare: double]
val res15: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Pclass: int, Average_Fare: double]
+------+------------+
|Pclass|Average_Fare|
+------+------------+
|     1|       84.15|
|     2|       20.66|
|     3|       13.68|
+------+------------+
```



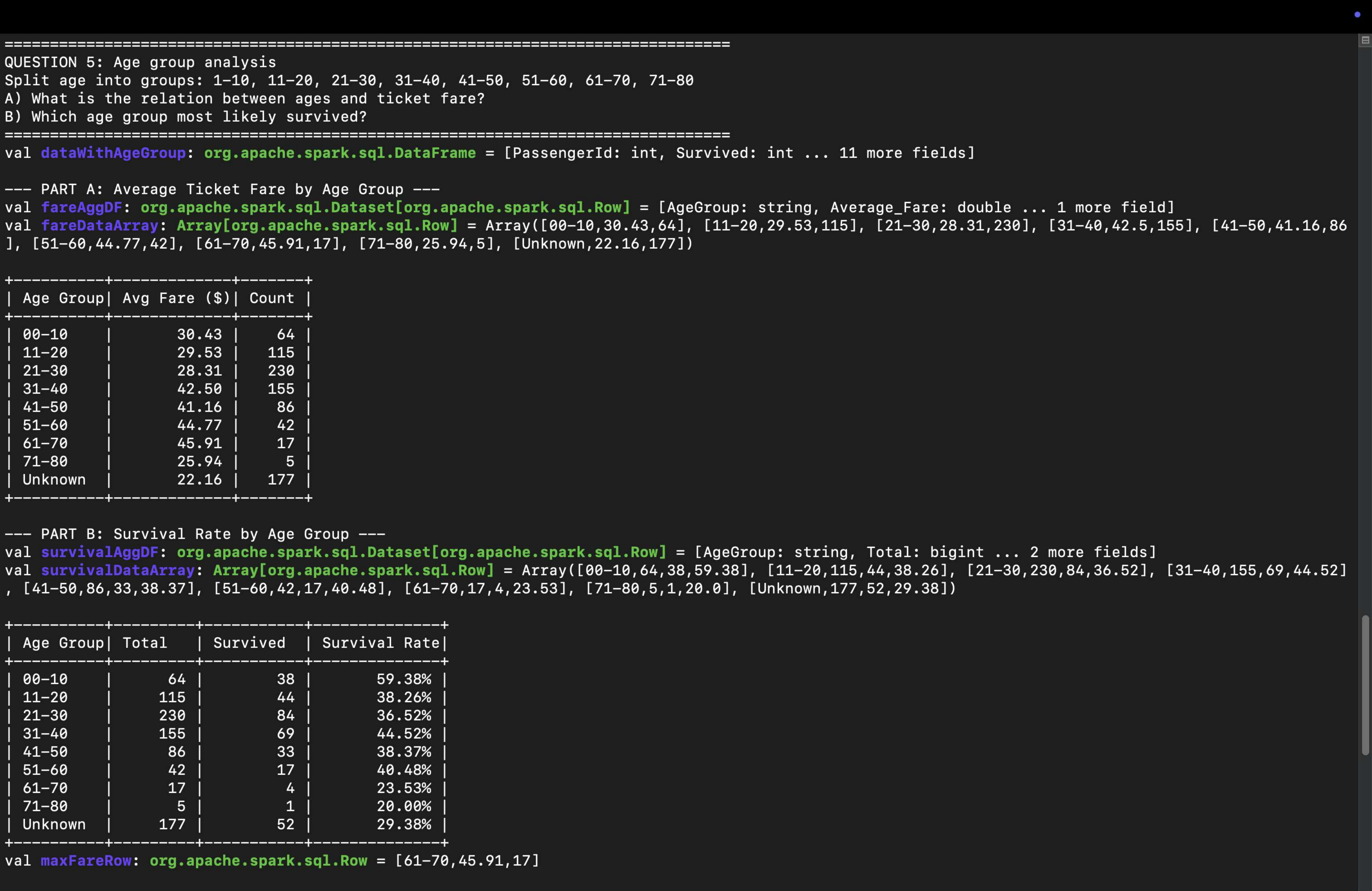
```
+------+------------+
|Pclass|Average_Fare|
+------+------------+
|     1|       84.15|
|     2|       20.66|
|     3|       13.68|
+------+------------+

val fare1Result: Double = 84.15
val fare2Result: Double = 20.66
val fare3Result: Double = 13.68
ANSWER TO QUESTION 1:
1st Class (Upper): $84.15 average fare
2nd Class (Middle): $20.66 average fare
3rd Class (Lower): $13.68 average fare

================================================================================
QUESTION 2: What is the survival percentage for each Ticket class?
Which class has the highest survival rate?
================================================================================

Survival Statistics by Class:
val res26: org.apache.spark.sql.RelationalGroupedDataset = RelationalGroupedDataset: [grouping expressions: [Pclass: int], value: [PassengerId: int,
 Survived: int ... 10 more fields], type: GroupBy]
val res27: org.apache.spark.sql.DataFrame = [Pclass: int, Total: bigint ... 2 more fields]
val res28: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Pclass: int, Total: bigint ... 2 more fields]
+------+-----+---------+-------------+
|Pclass|Total|Survivors|Survival_Rate|
+------+-----+---------+-------------+
|     1|  216|      136|        62.96|
|     2|  184|       87|        47.28|
|     3|  491|      119|        24.24|
+------+-----+---------+-------------+

val survival1: Double = 62.96
val survival2: Double = 47.28
val survival3: Double = 24.24
val bestClassNum: Int = 1
val bestClassRate: Double = 62.96
ANSWER TO QUESTION 2:
1st Class (Upper): 62.96% survival rate
2nd Class (Middle): 47.28% survival rate
3rd Class (Lower): 24.24% survival rate
Class 1 has the HIGHEST survival rate at 62.96%

================================================================================
```



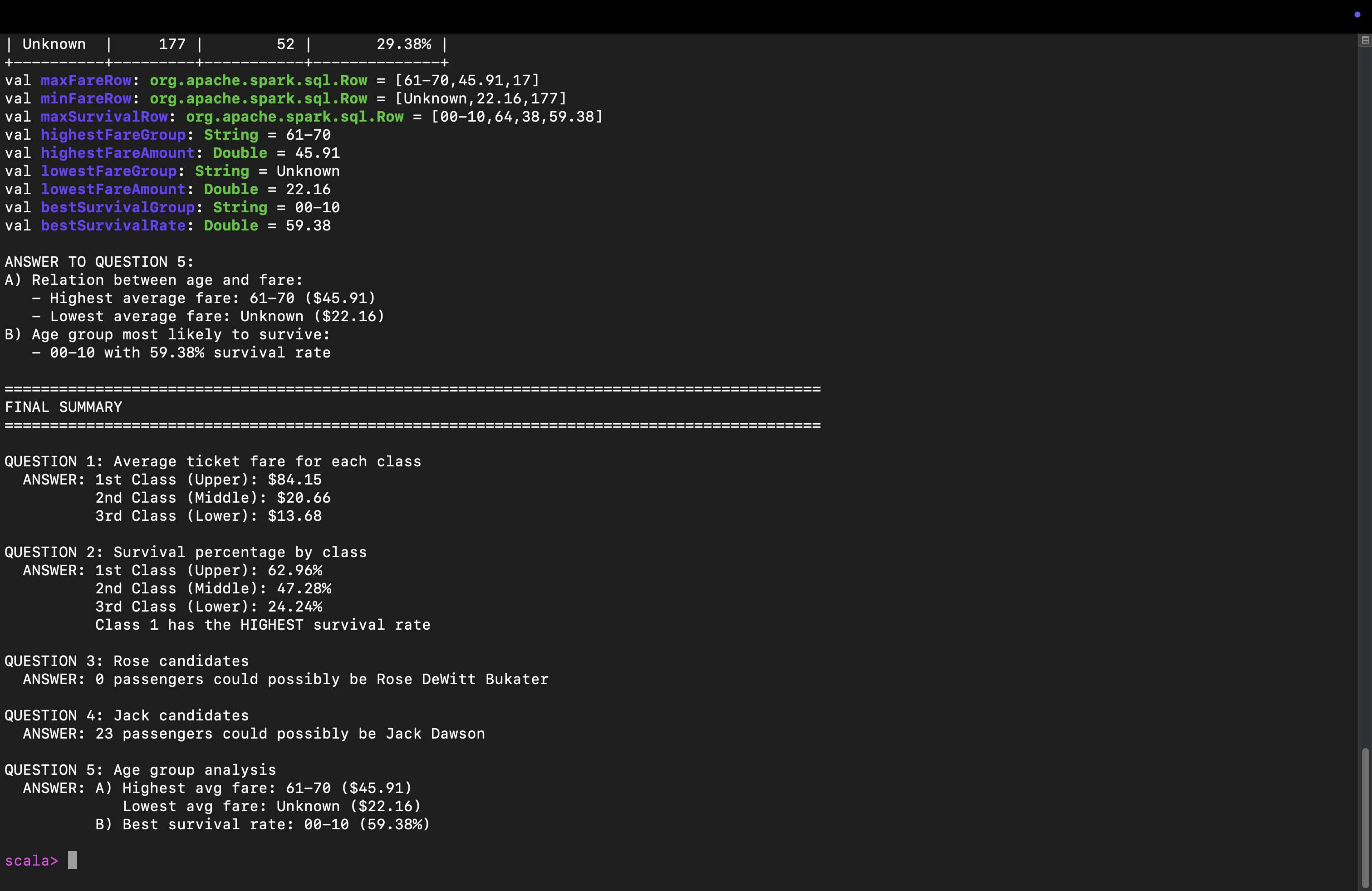
```
+------+-----+---------+-------------+
|Pclass|Total|Survivors|Survival_Rate|
+------+-----+---------+-------------+
|     1|  216|      136|        62.96|
|     2|  184|       87|        47.28|
|     3|  491|      119|        24.24|
+------+-----+---------+-------------+

val survival1: Double = 62.96
val survival2: Double = 47.28
val survival3: Double = 24.24
val bestClassNum: Int = 1
val bestClassRate: Double = 62.96
ANSWER TO QUESTION 2:
1st Class (Upper): 62.96% survival rate
2nd Class (Middle): 47.28% survival rate
3rd Class (Lower): 24.24% survival rate
Class 1 has the HIGHEST survival rate at 62.96%

================================================================================
QUESTION 3: Find passengers who could possibly be Rose DeWitt Bukater
Rose's characteristics:
- Age: 17 years old
- Gender: Female
- Class: 1st Class
- Traveling with: 1 parent (Parch = 1)
================================================================================
val rose: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [PassengerId: int, Survived: int ... 10 more fields]
val roseCount: Long = 0

Number of passengers who could possibly be Rose: 0
No exact matches found for Rose's characteristics.

ANSWER TO QUESTION 3: 0 passengers could possibly be Rose

================================================================================
QUESTION 4: Find passengers who could possibly be Jack Dawson
Jack's characteristics:
- Born: 1892, Died: April 15, 1912
- Age: 19 or 20 years old
- Gender: Male
- Class: 3rd Class
- No relatives onboard (SibSp = 0, Parch = 0)
================================================================================
val jack: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [PassengerId: int, Survived: int ... 10 more fields]
val jackCount: Long = 23
```



```
QUESTION 4: Find passengers who could possibly be Jack Dawson
Jack's characteristics:
- Born: 1892, Died: April 15, 1912
- Age: 19 or 20 years old
- Gender: Male
- Class: 3rd Class
- No relatives onboard (SibSp = 0, Parch = 0)
================================================================================
val jack: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [PassengerId: int, Survived: int ... 10 more fields]
val jackCount: Long = 23

Number of passengers who could possibly be Jack: 23

Possible Jack candidates:
+-----------+------------------------------------------+----+----+------+-----+-----+--------+
|PassengerId|Name                                      |Age |Sex |Pclass|SibSp|Parch|Survived|
+-----------+------------------------------------------+----+----+------+-----+-----+--------+
|13         |Saundercock, Mr. William Henry            |20.0|male|3     |0    |0    |0       |
|68         |Crease, Mr. Ernest James                  |19.0|male|3     |0    |0    |0       |
|92         |Andreasson, Mr. Paul Edvin                |20.0|male|3     |0    |0    |0       |
|132        |Coelho, Mr. Domingos Fernandeo            |20.0|male|3     |0    |0    |0       |
|144        |Burke, Mr. Jeremiah                       |19.0|male|3     |0    |0    |0       |
|284        |Dorking, Mr. Edward Arthur                |19.0|male|3     |0    |0    |1       |
|303        |Johnson, Mr. William Cahoone Jr           |19.0|male|3     |0    |0    |0       |
|373        |Beavan, Mr. William Thomas                |19.0|male|3     |0    |0    |0       |
|379        |Betros, Mr. Tannous                       |20.0|male|3     |0    |0    |0       |
|380        |Gustafsson, Mr. Karl Gideon               |19.0|male|3     |0    |0    |0       |
|442        |Hampe, Mr. Leon                           |20.0|male|3     |0    |0    |0       |
|567        |Stoytcheff, Mr. Ilia                      |19.0|male|3     |0    |0    |0       |
|576        |Patchett, Mr. George                      |19.0|male|3     |0    |0    |0       |
|641        |Jensen, Mr. Hans Peder                    |20.0|male|3     |0    |0    |0       |
|647        |Cor, Mr. Liudevit                         |19.0|male|3     |0    |0    |0       |
|683        |Olsvigen, Mr. Thor Anderson               |20.0|male|3     |0    |0    |0       |
|688        |Dakic, Mr. Branko                         |19.0|male|3     |0    |0    |0       |
|716        |Soholt, Mr. Peter Andreas Lauritz Andersen|19.0|male|3     |0    |0    |0       |
|726        |Oreskovic, Mr. Luka                       |20.0|male|3     |0    |0    |0       |
|763        |Barah, Mr. Hanna Assi                     |20.0|male|3     |0    |0    |1       |
|841        |Alhomaki, Mr. Ilmari Rudolf               |20.0|male|3     |0    |0    |0       |
|877        |Gustafsson, Mr. Alfred Ossian             |20.0|male|3     |0    |0    |0       |
|878        |Petroff, Mr. Nedelio                      |19.0|male|3     |0    |0    |0       |
+-----------+------------------------------------------+----+----+------+-----+-----+--------+

Survival: 2 survived, 21 did not survive
Found matches for people who can be jack

ANSWER TO QUESTION 4: 23 passengers could possibly be Jack
```



```
================================================================================
QUESTION 5: Age group analysis
Split age into groups: 1-10, 11-20, 21-30, 31-40, 41-50, 51-60, 61-70, 71-80
A) What is the relation between ages and ticket fare?
B) Which age group most likely survived?
================================================================================
val dataWithAgeGroup: org.apache.spark.sql.DataFrame = [PassengerId: int, Survived: int ... 11 more fields]

--- PART A: Average Ticket Fare by Age Group ---
val fareAggDF: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [AgeGroup: string, Average_Fare: double ... 1 more field]
val fareDataArray: Array[org.apache.spark.sql.Row] = Array([00-10,30.43,64], [11-20,29.53,115], [21-30,28.31,230], [31-40,42.5,155], [41-50,41.16,86
], [51-60,44.77,42], [61-70,45.91,17], [71-80,25.94,5], [Unknown,22.16,177])

+----------+-------------+-------+
| Age Group| Avg Fare ($)| Count |
+----------+-------------+-------+
| 00-10    |       30.43 |    64 |
| 11-20    |       29.53 |   115 |
| 21-30    |       28.31 |   230 |
| 31-40    |       42.50 |   155 |
| 41-50    |       41.16 |    86 |
| 51-60    |       44.77 |    42 |
| 61-70    |       45.91 |    17 |
| 71-80    |       25.94 |     5 |
| Unknown  |       22.16 |   177 |
+----------+-------------+-------+

--- PART B: Survival Rate by Age Group ---
val survivalAggDF: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [AgeGroup: string, Total: bigint ... 2 more fields]
val survivalDataArray: Array[org.apache.spark.sql.Row] = Array([00-10,64,38,59.38], [11-20,115,44,38.26], [21-30,230,84,36.52], [31-40,155,69,44.52]
, [41-50,86,33,38.37], [51-60,42,17,40.48], [61-70,17,4,23.53], [71-80,5,1,20.0], [Unknown,177,52,29.38])

+----------+---------+-----------+--------------+
| Age Group| Total   | Survived  | Survival Rate|
+----------+---------+-----------+--------------+
| 00-10    |      64 |        38 |       59.38% |
| 11-20    |     115 |        44 |       38.26% |
| 21-30    |     230 |        84 |       36.52% |
| 31-40    |     155 |        69 |       44.52% |
| 41-50    |      86 |        33 |       38.37% |
| 51-60    |      42 |        17 |       40.48% |
| 61-70    |      17 |         4 |       23.53% |
| 71-80    |       5 |         1 |       20.00% |
| Unknown  |     177 |        52 |       29.38% |
+----------+---------+-----------+--------------+
val maxFareRow: org.apache.spark.sql.Row = [61-70,45.91,17]
```



```
| Unknown  |      177 |        52 |       29.38% |
+----------+----------+------------+--------------+
val maxFareRow: org.apache.spark.sql.Row = [61-70,45.91,17]
val minFareRow: org.apache.spark.sql.Row = [Unknown,22.16,177]
val maxSurvivalRow: org.apache.spark.sql.Row = [00-10,64,38,59.38]
val highestFareGroup: String = 61-70
val highestFareAmount: Double = 45.91
val lowestFareGroup: String = Unknown
val lowestFareAmount: Double = 22.16
val bestSurvivalGroup: String = 00-10
val bestSurvivalRate: Double = 59.38

ANSWER TO QUESTION 5:
A) Relation between age and fare:
   - Highest average fare: 61-70 ($45.91)
   - Lowest average fare: Unknown ($22.16)
B) Age group most likely to survive:
   - 00-10 with 59.38% survival rate

==========================================================================================
FINAL SUMMARY
==========================================================================================

QUESTION 1: Average ticket fare for each class
  ANSWER: 1st Class (Upper): $84.15
          2nd Class (Middle): $20.66
          3rd Class (Lower): $13.68

QUESTION 2: Survival percentage by class
  ANSWER: 1st Class (Upper): 62.96%
          2nd Class (Middle): 47.28%
          3rd Class (Lower): 24.24%
          Class 1 has the HIGHEST survival rate

QUESTION 3: Rose candidates
  ANSWER: 0 passengers could possibly be Rose DeWitt Bukater

QUESTION 4: Jack candidates
  ANSWER: 23 passengers could possibly be Jack Dawson

QUESTION 5: Age group analysis
  ANSWER: A) Highest avg fare: 61-70 ($45.91)
             Lowest avg fare: Unknown ($22.16)
          B) Best survival rate: 00-10 (59.38%)

scala> 
```