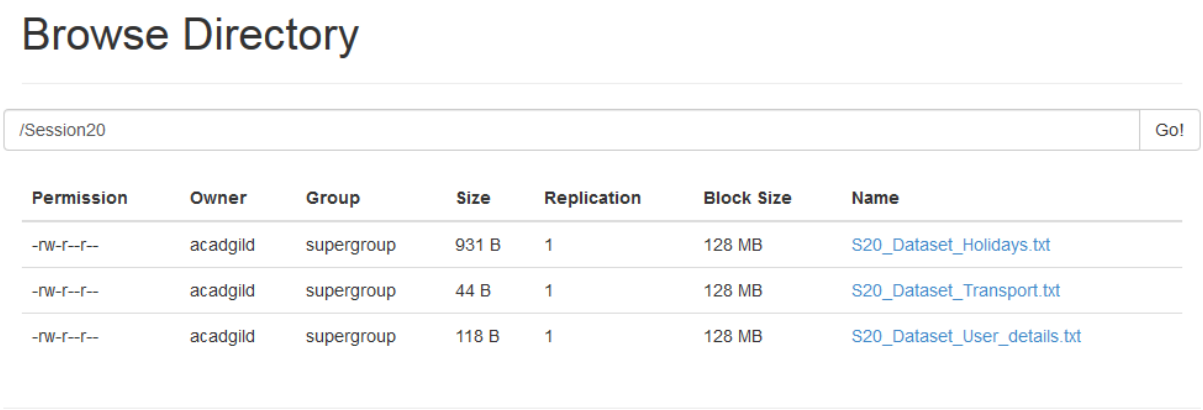

Task 1

1) What is the distribution of the total number of air-travelers per year

```
scala> val HolidaySQL = HolidayDF.toDF()
HolidaySQL: org.apache.spark.sql.DataFrame = [id: int, source: string ... 4 more
 fields]

scala> HolidaySQL.show()
+---+------+----+--------+----+----+
| id|source|dest|    mode|dist|year|
+---+------+----+--------+----+----+
|  1|   CHN| IND|airplane| 200|1990|
|  2|   IND| CHN|airplane| 200|1991|
|  3|   IND| CHN|airplane| 200|1992|
|  4|   RUS| IND|airplane| 200|1990|
|  5|   CHN| RUS|airplane| 200|1992|
|  6|   AUS| PAK|airplane| 200|1991|
|  7|   RUS| AUS|airplane| 200|1990|
|  8|   IND| RUS|airplane| 200|1991|
|  9|   CHN| RUS|airplane| 200|1992|
| 10|   AUS| CHN|airplane| 200|1993|
|  1|   AUS| CHN|airplane| 200|1993|
|  2|   CHN| IND|airplane| 200|1993|
|  3|   CHN| IND|airplane| 200|1993|
|  4|   IND| AUS|airplane| 200|1991|
|  5|   AUS| IND|airplane| 200|1992|
|  6|   RUS| CHN|airplane| 200|1993|
|  7|   CHN| RUS|airplane| 200|1990|
|  8|   AUS| CHN|airplane| 200|1990|
|  9|   IND| AUS|airplane| 200|1991|
| 10|   RUS| CHN|airplane| 200|1992|
+---+------+----+--------+----+----+
only showing top 20 rows
```

```
scala> import org.apache.spark.sql.catalyst.encoders.ExpressionEncoder
import org.apache.spark.sql.catalyst.encoders.ExpressionEncoder

scala> import org.apache.spark.sql.Encoder
import org.apache.spark.sql.Encoder

scala> import org.apache.spark.sql.{Row, SparkSession}
import org.apache.spark.sql.{Row, SparkSession}

scala> import org.apache.spark.sql.types.{DoubleType, StringType, StructField, S
tructType}
import org.apache.spark.sql.types.{DoubleType, StringType, StructField, StructTy
pe}

scala> import spark.implicits._
import spark.implicits._

scala>

scala> case class Holiday(id:Int, source:String, dest:String, mode:String, dist:
Int, year:Int)
defined class Holiday

scala>

scala> val HolidayFile = sc.textFile("/Session20/S20_Dataset_Holidays.txt")
HolidayFile: org.apache.spark.rdd.RDD[String] = /Session20/S20_Dataset_Holidays.
txt MapPartitionsRDD[12] at textFile at <console>:40

scala>

scala> val HolidayDF = HolidayFile.map(_.split(",")).map(attributes => Holiday(a
ttributes(0).toInt,attributes(1),attributes(2),attributes(3),attributes(4).toInt
,attributes(5).trim.toInt))
HolidayDF: org.apache.spark.rdd.RDD[Holiday] = MapPartitionsRDD[14] at map at <c
onsole>:44

scala>

scala> val HolidaySQL = HolidayDF.toDF()
HolidaySQL: org.apache.spark.sql.DataFrame = [id: int, source: string ... 4 more
 fields]

scala>

scala> val AirDistrib = HolidaySQL.groupBy("year").count()
AirDistrib: org.apache.spark.sql.DataFrame = [year: int, count: bigint]

scala> AirDistrib.show()
+----+-----+
|year|count|
+----+-----+
|1990|    8|
|1994|    1|
|1991|    9|
|1992|    7|
|1993|    7|
+----+-----+
```

2) What is the total air distance covered by each user per year

```
scala> val AirDistByUser = HolidaySQL.groupBy("id","year").agg(sum("dist"))
AirDistByUser: org.apache.spark.sql.DataFrame = [id: int, year: int ... 1 more f
ield]

scala> AirDistByUser.show()
+---+----+---------+
| id|year|sum(dist)|
+---+----+---------+
|  3|1991|      200|
|  6|1993|      200|
|  3|1992|      200|
|  7|1990|      600|
| 10|1993|      200|
|  6|1991|      400|
|  2|1991|      400|
|  4|1991|      200|
|  5|1991|      200|
|  5|1994|      200|
|  8|1991|      200|
|  1|1990|      200|
|  5|1992|      400|
|  4|1990|      400|
|  3|1993|      200|
| 10|1990|      200|
|  2|1993|      200|
|  1|1993|      600|
|  9|1991|      200|
|  9|1992|      400|
+---+----+---------+
only showing top 20 rows
```

3) Which user has travelled the largest distance till date

```
scala> val UserMaxDis = HolidaySQL.groupBy("id").agg(sum("dist")).sort(desc("sum
(dist)")).show(1)
+---+---------+
| id|sum(dist)|
+---+---------+
|  1|      800|
+---+---------+
only showing top 1 row

UserMaxDis: Unit = ()

scala>
```

4) What is the most preferred destination for all users.

Preferred destination is: IND

```
scala> val FavDist = HolidaySQL.groupBy("dest").count().sort(desc("count")).show
(1)
+----+-----+
|dest|count|
+----+-----+
| IND|    9|
+----+-----+
only showing top 1 row

FavDist: Unit = ()
```

```
scala> val DistAll = HolidaySQL.groupBy("dest").count().sort(desc("count")).show
()
+----+-----+
|dest|count|
+----+-----+
| IND|    9|
| CHN|    7|
| RUS|    6|
| AUS|    5|
| PAK|    5|
+----+-----+

DistAll: Unit = ()

scala> |
```

5)Which route is generating the most revenue per year

```
scala> val HighRevRouteAll = HolidaySQL.groupBy("year","source","dest").agg(sum(
"dist")).sort(desc("sum(dist)")).show()
+----+------+----+---------+
|year|source|dest|sum(dist)|
+----+------+----+---------+
|1991|   IND| RUS|      400|
|1993|   AUS| CHN|      400|
|1990|   CHN| IND|      400|
|1992|   RUS| IND|      400|
|1991|   IND| AUS|      400|
|1993|   CHN| IND|      400|
|1992|   CHN| RUS|      400|
|1994|   CHN| PAK|      200|
|1993|   PAK| AUS|      200|
|1993|   PAK| IND|      200|
|1990|   CHN| PAK|      200|
|1990|   CHN| RUS|      200|
|1991|   IND| CHN|      200|
|1992|   RUS| CHN|      200|
|1991|   AUS| PAK|      200|
|1992|   IND| CHN|      200|
|1992|   AUS| IND|      200|
|1991|   CHN| PAK|      200|
|1990|   CHN| AUS|      200|
|1990|   RUS| AUS|      200|
+----+------+----+---------+
only showing top 20 rows

HighRevRouteAll: Unit = ()
```

6) What is the total amount spent by every user on air-travel per year

```
scala> TransSQL.show()
+--------+----+
|    mode|cost|
+--------+----+
|airplane| 170|
|     car| 140|
|   train| 120|
|    ship| 200|
+--------+----+


scala> UserSQL.show()
+---+------+---+
| id|  name|age|
+---+------+---+
|  1|  mark| 15|
|  2|  john| 16|
|  3|  luke| 17|
|  4|  lisa| 27|
|  5|  mark| 25|
|  6| peter| 22|
|  7| james| 21|
|  8|andrew| 55|
|  9|thomas| 46|
| 10| annie| 44|
+---+------+---+


scala> HolidaySQL.show()
+---+------+----+--------+----+----+
| id|source|dest|    mode|dist|year|
+---+------+----+--------+----+----+
|  1|   CHN| IND|airplane| 200|1990|
|  2|   IND| CHN|airplane| 200|1991|
|  3|   IND| CHN|airplane| 200|1992|
|  4|   RUS| IND|airplane| 200|1990|
|  5|   CHN| RUS|airplane| 200|1992|
|  6|   AUS| PAK|airplane| 200|1991|
|  7|   RUS| AUS|airplane| 200|1990|
|  8|   IND| RUS|airplane| 200|1991|
|  9|   CHN| RUS|airplane| 200|1992|
| 10|   AUS| CHN|airplane| 200|1993|
|  1|   AUS| CHN|airplane| 200|1993|
|  2|   CHN| IND|airplane| 200|1993|
|  3|   CHN| IND|airplane| 200|1993|
|  4|   IND| AUS|airplane| 200|1991|
|  5|   AUS| IND|airplane| 200|1992|
|  6|   RUS| CHN|airplane| 200|1993|
|  7|   CHN| RUS|airplane| 200|1990|
|  8|   AUS| CHN|airplane| 200|1990|
|  9|   IND| AUS|airplane| 200|1991|
| 10|   RUS| CHN|airplane| 200|1992|
+---+------+----+--------+----+----+
only showing top 20 rows
```

```
scala> val travelCost = HolidaySQL.join(TransSQL, "mode")
travelCost: org.apache.spark.sql.DataFrame = [mode: string, id: int ... 5 more f
ields]

scala> travelCost.show()
+--------+---+------+----+----+----+----+
|    mode| id|source|dest|dist|year|cost|
+--------+---+------+----+----+----+----+
|airplane|  1|   CHN| IND| 200|1990| 170|
|airplane|  2|   IND| CHN| 200|1991| 170|
|airplane|  3|   IND| CHN| 200|1992| 170|
|airplane|  4|   RUS| IND| 200|1990| 170|
|airplane|  5|   CHN| RUS| 200|1992| 170|
|airplane|  6|   AUS| PAK| 200|1991| 170|
|airplane|  7|   RUS| AUS| 200|1990| 170|
|airplane|  8|   IND| RUS| 200|1991| 170|
|airplane|  9|   CHN| RUS| 200|1992| 170|
|airplane| 10|   AUS| CHN| 200|1993| 170|
|airplane|  1|   AUS| CHN| 200|1993| 170|
|airplane|  2|   CHN| IND| 200|1993| 170|
|airplane|  3|   CHN| IND| 200|1993| 170|
|airplane|  4|   IND| AUS| 200|1991| 170|
|airplane|  5|   AUS| IND| 200|1992| 170|
|airplane|  6|   RUS| CHN| 200|1993| 170|
|airplane|  7|   CHN| RUS| 200|1990| 170|
|airplane|  8|   AUS| CHN| 200|1990| 170|
|airplane|  9|   IND| AUS| 200|1991| 170|
|airplane| 10|   RUS| CHN| 200|1992| 170|
+--------+---+------+----+----+----+----+
only showing top 20 rows


scala> val costByuser = travelCost.groupBy("year","id").agg(sum("cost")).show()
+----+---+---------+
|year| id|sum(cost)|
+----+---+---------+
|1990|  7|      510|
|1990|  4|      340|
|1991|  5|      170|
|1993|  3|      170|
|1990|  1|      170|
|1990|  8|      170|
|1991|  9|      170|
|1992|  3|      170|
|1993|  1|      510|
|1993|  2|      170|
|1993|  6|      170|
|1991|  8|      170|
|1993| 10|      170|
|1992| 10|      170|
|1990| 10|      170|
|1994|  5|      170|
|1992|  5|      340|
|1991|  6|      340|
|1991|  2|      340|
|1992|  8|      170|
+----+---+---------+
only showing top 20 rows

costByuser: Unit = ()

scala>
```

7) Considering age groups of < 20 , 20-35, 35 > ,Which age group is travelling the most every year.

```
scala> UserSQL.show()
+---+------+---+
| id|  name|age|
+---+------+---+
|  1|  mark| 15|
|  2|  john| 16|
|  3|  luke| 17|
|  4|  lisa| 27|
|  5|  mark| 25|
|  6| peter| 22|
|  7| james| 21|
|  8|andrew| 55|
|  9|thomas| 46|
| 10| annie| 44|
+---+------+---+


scala> travelCost.show()
+--------+---+------+----+----+----+----+
|    mode| id|source|dest|dist|year|cost|
+--------+---+------+----+----+----+----+
|airplane|  1|   CHN| IND| 200|1990| 170|
|airplane|  2|   IND| CHN| 200|1991| 170|
|airplane|  3|   IND| CHN| 200|1992| 170|
|airplane|  4|   RUS| IND| 200|1990| 170|
|airplane|  5|   CHN| RUS| 200|1992| 170|
|airplane|  6|   AUS| PAK| 200|1991| 170|
|airplane|  7|   RUS| AUS| 200|1990| 170|
|airplane|  8|   IND| RUS| 200|1991| 170|
|airplane|  9|   CHN| RUS| 200|1992| 170|
|airplane| 10|   AUS| CHN| 200|1993| 170|
|airplane|  1|   AUS| CHN| 200|1993| 170|
|airplane|  2|   CHN| IND| 200|1993| 170|
|airplane|  3|   CHN| IND| 200|1993| 170|
|airplane|  4|   IND| AUS| 200|1991| 170|
|airplane|  5|   AUS| IND| 200|1992| 170|
|airplane|  6|   RUS| CHN| 200|1993| 170|
|airplane|  7|   CHN| RUS| 200|1990| 170|
|airplane|  8|   AUS| CHN| 200|1990| 170|
|airplane|  9|   IND| AUS| 200|1991| 170|
|airplane| 10|   RUS| CHN| 200|1992| 170|
+--------+---+------+----+----+----+----+
only showing top 20 rows
```

```
scala> val UserAgeSQL = travelCost.join(UserSQL, "id")
UserAgeSQL: org.apache.spark.sql.DataFrame = [id: int, mode: string ... 7 more f
ields]

scala> UserAgeSQL.show()
+---+--------+------+----+----+----+----+------+---+
| id|    mode|source|dest|dist|year|cost|  name|age|
+---+--------+------+----+----+----+----+------+---+
|  1|airplane|   CHN| IND| 200|1990| 170|  mark| 15|
|  1|airplane|   AUS| CHN| 200|1993| 170|  mark| 15|
|  1|airplane|   PAK| IND| 200|1993| 170|  mark| 15|
|  1|airplane|   PAK| AUS| 200|1993| 170|  mark| 15|
|  6|airplane|   AUS| PAK| 200|1991| 170| peter| 22|
|  6|airplane|   RUS| CHN| 200|1993| 170| peter| 22|
|  6|airplane|   PAK| RUS| 200|1991| 170| peter| 22|
|  3|airplane|   IND| CHN| 200|1992| 170|  luke| 17|
|  3|airplane|   CHN| IND| 200|1993| 170|  luke| 17|
|  3|airplane|   CHN| PAK| 200|1991| 170|  luke| 17|
|  5|airplane|   CHN| RUS| 200|1992| 170|  mark| 25|
|  5|airplane|   AUS| IND| 200|1992| 170|  mark| 25|
|  5|airplane|   IND| PAK| 200|1991| 170|  mark| 25|
|  5|airplane|   CHN| PAK| 200|1994| 170|  mark| 25|
|  9|airplane|   CHN| RUS| 200|1992| 170|thomas| 46|
|  9|airplane|   IND| AUS| 200|1991| 170|thomas| 46|
|  9|airplane|   RUS| IND| 200|1992| 170|thomas| 46|
|  4|airplane|   RUS| IND| 200|1990| 170|  lisa| 27|
|  4|airplane|   IND| AUS| 200|1991| 170|  lisa| 27|
|  4|airplane|   CHN| PAK| 200|1990| 170|  lisa| 27|
+---+--------+------+----+----+----+----+------+---+
only showing top 20 rows


scala>
```

Creating UDF

```
scala> val AgeCat = udf((age: Int) => ( if (age < 20) { "Young" } else if ((age
>= 20) && (age <=35)) { "Middle Age" } else if(age > 35) { "Senior" } else " " )
)
AgeCat: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFuncti
on(<function1>,StringType,Some(List(IntegerType)))

scala>
```

Results

```
scala> val GroupAgeCat = UserAgeSQLCat.groupBy("year","Category").count().sort("
year","Category").show()
+----+----------+-----+
|year|  Category|count|
+----+----------+-----+
|1990|Middle Age|    5|
|1990|    Senior|    2|
|1990|     Young|    1|
|1991|Middle Age|    4|
|1991|    Senior|    2|
|1991|     Young|    3|
|1992|Middle Age|    2|
|1992|    Senior|    4|
|1992|     Young|    1|
|1993|Middle Age|    1|
|1993|    Senior|    1|
|1993|     Young|    5|
|1994|Middle Age|    1|
+----+----------+-----+

GroupAgeCat: Unit = ()

scala>
```