***A.*** *Data retrieval using Spark SQL*

1. Retrieve all Driver data (use Drivers.CSV)

* dfDrivers.show()

2. Retrieve all Taxis and display the data in ascending order of Taxi License Plate number.

* dfTaxiCabs.orderBy(asc("TaxiNumber")).show()

3. Retrieve all Limousine Taxies. You should display only the Taxi Number, Taxi Type, and Taxi Colour.

* ( dfTaxiCabs.select("TaxiNumber", "TaxiType", "TaxiColor").where(col("TaxiType") == "Limosine").show() )

4. Retrieve all 4 seater Premier taxis.

* ( dfTaxiCabs.where((col("TaxiPassengerCapacity") == 4) & (col("TaxiType") == "Premier")).show() )

***B.*** *Aggregation and Statistical Queries (use BEAD\_Rebu\_TripData.CSV)*

5. Determine the **average distance per trip** based on ALL trips in the month of January 2024.

* dfTripData.agg(avg("Distance Travelled")).first()[0]

6. Find the total fares collected grouped by Taxi Type **Maxi Cab**

* dfTripData.groupBy("Taxi Type").sum("Trip Fare").where(col("Taxi Type") == "Maxi Cab").show()

***C.*** *Analytics Questions*

7. Determine the **Average Occupancy** i.e., *(Number of Passengers / Passenger Capacity)* for Standard Taxis.

* dfStandardTaxis = dfTripData.where(col("Taxi Type") == "Standard")
* dfStandardTaxis.agg(avg("Number Of Passengers")).first()[0]

8. Determine **Fares Collected by Day of the Week** (ie., Sun, Mon, Tue) for the month of Jan 2024.

* dfTripData.where(col("Date").like("%Jan%")).groupBy("Day").sum("Trip Fare").show()

9. Prepare a **Tabulation report showing total revenue** against the two dimensions *Hour of the day* AND *Day of the Week.*

* dfTripData.rollup("Hour Of Day", "Day").sum("Trip Fare").orderBy("Hour Of Day", "Day").show()

10. Compare the *total number of trips per day made by all taxis in weekends* **vs** *the total number of trips made per day during weekdays* in the month of Jan 2024.

* dfTripData.cube("Hour Of Day", "Day").sum("Distance Travelled").orderBy("Hour Of Day", "Day").show()

***D.*** *Multiple Entities Joining and multiple formats joined in a DataFrame*

11. Determine the **total fares paid by all Gold Status Passengers** in the month of Jan 2024. What percentage does this make from the total fares for all customers in month of Jan 2024.

* dfPassengersNewHdr = dfPassengers.select(col("PassengerID").alias("Passenger ID"), col("MemSilvererStGoldtus") )
* combineDf = dfTripData.join(dfPassengersNewHdr, "Passenger ID")
* totalFares = combineDf.agg(sum("Trip Fare"))
* totalFaresGold = combineDf.filter(col("MemSilvererStGoldtus") == "Gold").agg(sum("Trip Fare"))
* results = (totalFaresGold.collect()[0][0] / totalFares.collect()[0][0]) \* 100
* print(f"Results: {results}%")