Task 1 - SQL

Build SparkSession:

In [1]:
```
import findspark
import pyspark

findspark.init()
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('Assignment 2').getOrCreate()
```

Read the json file:

In [2]:
```
df = spark.read.json('DataFrames_sample.json',)
```

Display the schema:

In [3]:
```
df.printSchema()
```

```
root
 |-- D: double (nullable = true)
 |-- H: double (nullable = true)
 |-- HDD: string (nullable = true)
 |-- Id: long (nullable = true)
 |-- Model: string (nullable = true)
 |-- RAM: string (nullable = true)
 |-- ScreenSize: string (nullable = true)
 |-- W: double (nullable = true)
 |-- Weight: double (nullable = true)
 |-- Year: long (nullable = true)
```

Get all the data when "Model" equal "MacBook Pro":

```
In [4]: df.show()
```

```
+----+----+--------+---+-----------+----+----------+-----+------+----+
|   D|   H|     HDD| Id|      Model| RAM|ScreenSize|    W|Weight|Year|
+----+----+--------+---+-----------+----+----------+-----+------+----+
|9.48|0.61|512GB SSD|  1|MacBook Pro|16GB|       15"|13.75|  4.02|2015|
|7.74|0.52|256GB SSD|  2|    MacBook| 8GB|       12"|11.04|  2.03|2016|
|8.94|0.68|128GB SSD|  3|MacBook Air| 8GB|     13.3"| 12.8|  2.96|2016|
| 8.0|20.3|  1TB SSD|  4|       iMac|64GB|       27"| 25.6|  20.8|2017|
+----+----+--------+---+-----------+----+----------+-----+------+----+
```

```
In [5]: df.where((df["Model"] == 'MacBook Pro')).show()
```

```
+----+----+--------+---+-----------+----+----------+-----+------+----+
|   D|   H|     HDD| Id|      Model| RAM|ScreenSize|    W|Weight|Year|
+----+----+--------+---+-----------+----+----------+-----+------+----+
|9.48|0.61|512GB SSD|  1|MacBook Pro|16GB|       15"|13.75|  4.02|2015|
+----+----+--------+---+-----------+----+----------+-----+------+----+
```

Create TempView:

```
In [6]: df.createOrReplaceTempView("us_delay_flights_tbl")
```

Display "RAM"column and count "RAM" column:

```
In [9]: spark.sql("""SELECT RAM
                  FROM us_delay_flights_tbl""").show()
        spark.sql("""SELECT COUNT(RAM)
                  FROM us_delay_flights_tbl""").show()
```

```
+----+
| RAM|
+----+
|16GB|
| 8GB|
| 8GB|
|64GB|
+----+

+----------+
|count(RAM)|
+----------+
|         4|
+----------+
```

Get all columns when "Year" column equal "2015"

In [10]:
```
spark.sql("""SELECT *
          FROM us_delay_flights_tbl
          WHERE Year = 2015""").show()
```

```
+----+----+---------+---+-----------+----+----------+-----+------+----+
|   D|   H|      HDD| Id|      Model| RAM|ScreenSize|    W|Weight|Year|
+----+----+---------+---+-----------+----+----------+-----+------+----+
|9.48|0.61|512GB SSD|  1|MacBook Pro|16GB|       15"|13.75|  4.02|2015|
+----+----+---------+---+-----------+----+----------+-----+------+----+
```

Get all when "Model" start with "M":

In [12]:
```
spark.sql("""SELECT *
          FROM us_delay_flights_tbl
          WHERE Model LIKE 'M%'""").show()
```

```
+----+----+---------+---+-----------+----+----------+-----+------+----+
|   D|   H|      HDD| Id|      Model| RAM|ScreenSize|    W|Weight|Year|
+----+----+---------+---+-----------+----+----------+-----+------+----+
|9.48|0.61|512GB SSD|  1|MacBook Pro|16GB|       15"|13.75|  4.02|2015|
|7.74|0.52|256GB SSD|  2|    MacBook| 8GB|       12"|11.04|  2.03|2016|
|8.94|0.68|128GB SSD|  3|MacBook Air| 8GB|     13.3"| 12.8|  2.96|2016|
+----+----+---------+---+-----------+----+----------+-----+------+----+
```

Get all data when "Model" column equal "MacBook Pro"

In [13]:
```
spark.sql("""SELECT *
          FROM us_delay_flights_tbl
          WHERE Model = 'MacBook Pro'""").show()
```

```
+----+----+---------+---+-----------+----+----------+-----+------+----+
|   D|   H|      HDD| Id|      Model| RAM|ScreenSize|    W|Weight|Year|
+----+----+---------+---+-----------+----+----------+-----+------+----+
|9.48|0.61|512GB SSD|  1|MacBook Pro|16GB|       15"|13.75|  4.02|2015|
+----+----+---------+---+-----------+----+----------+-----+------+----+
```

Get all data with Multiple Conditions when "RAM" column equal "8GB" and "Model" column is "Macbook".

```
In [14]: spark.sql("""SELECT *
                   FROM us_delay_flights_tbl
                   WHERE RAM = '8GB' AND Model = 'MacBook'""").show()
```

```
+----+----+---------+---+-------+---+----------+-----+------+----+
|   D|   H|      HDD| Id|  Model|RAM|ScreenSize|    W|Weight|Year|
+----+----+---------+---+-------+---+----------+-----+------+----+
|7.74|0.52|256GB SSD|  2|MacBook|8GB|       12"|11.04|  2.03|2016|
+----+----+---------+---+-------+---+----------+-----+------+----+
```

Get all data with Multiple Conditions when "D" greater than or equal "8" and "Model" column is "iMac".

```
In [16]: spark.sql("""SELECT *
                   FROM us_delay_flights_tbl
                   WHERE D >= 8 AND Model = 'iMac'""").show()
```

```
+---+----+-------+---+-----+----+----------+----+------+----+
|  D|   H|    HDD| Id|Model| RAM|ScreenSize|   W|Weight|Year|
+---+----+-------+---+-----+----+----------+----+------+----+
|8.0|20.3|1TB SSD|  4| iMac|64GB|       27"|25.6|  20.8|2017|
+---+----+-------+---+-----+----+----------+----+------+----+
```

Task 2

Read "test1" dataset:

```
In [19]: test1 = spark.read.csv('test1.csv', header=True, inferSchema=True)
         test1.show()
```

```
+---------+---+----------+------+
|     Name|age|Experience|Salary|
+---------+---+----------+------+
|    Krish| 31|        10| 30000|
|Sudhanshu| 30|         8| 25000|
|    Sunny| 29|         4| 20000|
|     Paul| 24|         3| 20000|
|   Harsha| 21|         1| 15000|
|  Shubham| 23|         2| 18000|
+---------+---+----------+------+
```

Display Salary of the people less than or equal to 20000

In [20]:
```
test1.filter(test1['Salary'] <= 20000).select('Salary').show()
```

```
+------+
|Salary|
+------+
| 20000|
| 20000|
| 15000|
| 18000|
+------+
```

Display Salary of the people less than or equal to 20000 and greater than or equal 15000

In [21]:
```
test1.filter((test1['Salary'] <= 20000) & (test1['Salary'] >= 15000)).select(
'Salary').show()
```

```
+------+
|Salary|
+------+
| 20000|
| 20000|
| 15000|
| 18000|
+------+
```

Task 3

Read "test3" dataset:

In [23]:
```
from pyspark.sql.functions import *
test3 = spark.read.csv('test3.csv', header=True, inferSchema=True)
```

Display dataset

In [24]:
```
test3.show()
```

```
+---------+------------+------+
|     Name| Departments|salary|
+---------+------------+------+
|    Krish|Data Science| 10000|
|    Krish|         IOT|  5000|
|   Mahesh|    Big Data|  4000|
|    Krish|    Big Data|  4000|
|   Mahesh|Data Science|  3000|
|Sudhanshu|Data Science| 20000|
|Sudhanshu|         IOT| 10000|
|Sudhanshu|    Big Data|  5000|
|    Sunny|Data Science| 10000|
|    Sunny|    Big Data|  2000|
+---------+------------+------+
```

Display schema

In [25]:
```
test3.printSchema()
```

```
root
 |-- Name: string (nullable = true)
 |-- Departments: string (nullable = true)
 |-- salary: integer (nullable = true)
```

Group by "Name" column and using sum function on "Name" column

In [26]:
```
test3.groupBy('Name').sum().show()
```

```
+---------+-----------+
|     Name|sum(salary)|
+---------+-----------+
|Sudhanshu|      35000|
|    Sunny|      12000|
|    Krish|      19000|
|   Mahesh|       7000|
+---------+-----------+
```

Group by "Name" column and using avg function on "Name" column

```
In [27]: test3.groupBy('Name').avg().show()
```

```
+---------+------------------+
|     Name|       avg(salary)|
+---------+------------------+
|Sudhanshu|11666.666666666666|
|    Sunny|            6000.0|
|    Krish| 6333.333333333333|
|   Mahesh|            3500.0|
+---------+------------------+
```

Group by "Departments" column and using sum function on "Departments" column

```
In [28]: test3.groupBy('Departments').sum().show()
```

```
+------------+-----------+
| Departments|sum(salary)|
+------------+-----------+
|         IOT|      15000|
|    Big Data|      15000|
|Data Science|      43000|
+------------+-----------+
```

Group by "Departments" column and using mean function on "Departments" column:

```
In [29]: test3.groupBy('Departments').mean().show()
```

```
+------------+-----------+
| Departments|avg(salary)|
+------------+-----------+
|         IOT|     7500.0|
|    Big Data|     3750.0|
|Data Science|    10750.0|
+------------+-----------+
```

Group by "Departments" column and using count function on "Departments" column:

```
In [30]: test3.groupBy('Departments').count().show()
```

```
+------------+-----+
| Departments|count|
+------------+-----+
|         IOT|    2|
|    Big Data|    4|
|Data Science|    4|
+------------+-----+
```

Apply agg to using sum function get the total of salaries

```
In [32]: test3.select(sum('salary')).collect()[0][0]
```

```
Out[32]: 73000
```

Task 4

You've been flown to their headquarters in Ulsan, South Korea, to assist them in accurately estimating the number of crew members a ship will need.

They're currently building new ships for certain customers, and they'd like you to create a model and utilize it to estimate how many crew members the ships will require.

Metadata:

1. Measurements of ship size
2. capacity
3. crew
4. age for 158 cruise ships.

It is saved in a csv file for you called "ITI_data.csv". our task is to develop a regression model that will assist in predicting the number of crew members required for future ships. The client also indicated that they have found that particular cruise lines will differ in acceptable crew counts, thus this is most likely an important factor to consider when conducting your investigation.

In [36]:
```
data = spark.read.csv('ITI_data.csv', header=True, inferSchema=True)
data.show()
```

```
+-----------+-----------+---+------------------+----------+------+------+----
-------------+----+
|  Ship_name|Cruise_line|Age|          Tonnage|passengers|length|cabins|pass
enger_density|crew|
+-----------+-----------+---+------------------+----------+------+------+----
-------------+----+
|    Journey|    Azamara|  6|30.276999999999997|      6.94|  5.94|  3.55|
42.64|3.55|
|      Quest|    Azamara|  6|30.276999999999997|      6.94|  5.94|  3.55|
42.64|3.55|
|Celebration|   Carnival| 26|            47.262|     14.86|  7.22|  7.43|
31.8| 6.7|
|   Conquest|   Carnival| 11|             110.0|     29.74|  9.53| 14.88|
36.99|19.1|
|    Destiny|   Carnival| 17|           101.353|     26.42|  8.92| 13.21|
38.36|10.0|
|    Ecstasy|   Carnival| 22|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
|    Elation|   Carnival| 15|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
|    Fantasy|   Carnival| 23|            70.367|     20.56|  8.55| 10.22|
34.23| 9.2|
|Fascination|   Carnival| 19|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
|    Freedom|   Carnival|  6|110.23899999999999|      37.0|  9.51| 14.87|
29.79|11.5|
|      Glory|   Carnival| 10|             110.0|     29.74|  9.51| 14.87|
36.99|11.6|
|    Holiday|   Carnival| 28|            46.052|     14.52|  7.27|  7.26|
31.72| 6.6|
|Imagination|   Carnival| 18|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
|Inspiration|   Carnival| 17|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
|     Legend|   Carnival| 11|              86.0|     21.24|  9.63| 10.62|
40.49| 9.3|
|   Liberty*|   Carnival|  8|             110.0|     29.74|  9.51| 14.87|
36.99|11.6|
|    Miracle|   Carnival|  9|              88.5|     21.24|  9.63| 10.62|
41.67|10.3|
|   Paradise|   Carnival| 15|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
|      Pride|   Carnival| 12|              88.5|     21.24|  9.63| 11.62|
41.67| 9.3|
|  Sensation|   Carnival| 20|            70.367|     20.52|  8.55|  10.2|
34.29| 9.2|
+-----------+-----------+---+------------------+----------+------+------+----
-------------+----+
only showing top 20 rows
```

In [50]:
```
trainDF, testDF = data.randomSplit([.8,.2],seed=42)
```

In [51]:
```
data.columns
```

Out[51]:
```
['Ship_name',
 'Cruise_line',
 'Age',
 'Tonnage',
 'passengers',
 'length',
 'cabins',
 'passenger_density',
 'crew']
```

Use VectorAssembler to merge all columns into one column:

In [52]:
```
from pyspark.ml.feature import VectorAssembler
vecAssembler = VectorAssembler(inputCols=[
 'Age',
 'Tonnage',
 'passengers',
 'length',
 'cabins',
 'passenger_density',],outputCol='features')
```

Create a Linear Regression Model

In [53]:
```
from pyspark.ml.regression import LinearRegression
lr = LinearRegression(featuresCol='features',labelCol='crew')
```

Creating a Pipeline

In [54]:
```
from pyspark.ml import Pipeline
pipeline = Pipeline(stages=[vecAssembler,lr])
pipelineModel = pipeline.fit(trainDF)
```

Model Evaluation

In [56]:
```
from pyspark.ml.evaluation import RegressionEvaluator
regressionEvaluator = RegressionEvaluator(predictionCol='features',
                                          labelCol='crew',
                                          metricName='rmse')
rmse = regressionEvaluator.evaluate(testDF)
```

By Eng. Mostafa Nabieh If you have questions, please feel free to ask.

My Email : nabieh.mostafa@yahoo.com

My Whatsapp : +201015197566

In [ ]:

```

```