In [32]:
```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, isnan
```

Data profiling is the process of examining, analyzing, and creating useful summaries of data. ¶

In [33]:
```
# Initialize a Spark session
spark = SparkSession.builder.appName("DataProfiling").getOrCreate()
```

In [34]:
```
# Load your data into a Spark DataFrame
df = spark.read.csv("/Users/pragatigupta/Documents/AI And ML/Linkedin
```

In [35]:
```
df.head(30)
```

Out[35]:
```
[Row(ID=1, Student_ID='17975', Gender='F', AGE=15, Score=6.7, CLASS='
y '),
 Row(ID=2, Student_ID='34221', Gender='M', AGE=16, Score=6.5, CLASS='
y '),
 Row(ID=3, Student_ID='47975', Gender='F', AGE=17, Score=5.5, CLASS='
y '),
 Row(ID=4, Student_ID='87656', Gender='F', AGE=14, Score=6.8, CLASS='
y '),
 Row(ID=5, Student_ID='34223', Gender='M', AGE=15, Score=7.1, CLASS='
y '),
 Row(ID=6, Student_ID='34224', Gender='F', AGE=16, Score=2.3, CLASS='
N'),
 Row(ID=7, Student_ID='34225', Gender='F', AGE=17, Score=2.0, CLASS='
n'),
 Row(ID=8, Student_ID='34227', Gender='M', AGE=15, Score=4.7, CLASS='
N'),
 Row(ID=9, Student_ID='34229', Gender='M', AGE=16, Score=2.6, CLASS='
N '),
 Row(ID=10, Student_ID='34230', Gender='F', AGE=17, Score=6.7, CLASS=
'y '),
 Row(ID=11, Student_ID='34231', Gender='F', AGE=14, Score=6.5, CLASS=
'Y '),
 Row(ID=None, Student_ID='87656', Gender='F', AGE=14, Score=6.8, CLAS
S='y '),
 Row(ID=2, Student_ID='34221', Gender='M', AGE=16, Score=6.5, CLASS='
y '),
 Row(ID=14, Student_ID='34224', Gender='F', AGE=16, Score=2.3, CLASS=
'N'),
 Row(ID=15, Student_ID='34235', Gender='F', AGE=14, Score=3.5, CLASS=
'N'),
 Row(ID=16, Student_ID='34236', Gender='M', AGE=15, Score=5.5, CLASS=
'y '),
```

```
 Row(ID=17, Student_ID='34237', Gender='F', AGE=16, Score=5.9, CLASS=
'y '),
 Row(ID=18, Student_ID='87654', Gender='F', AGE=17, Score=6.7, CLASS=
'y '),
 Row(ID=19, Student_ID='34238', Gender='F', AGE=15, Score=6.5, CLASS=
'y '),
 Row(ID=20, Student_ID='34239', Gender='F', AGE=16, Score=5.5, CLASS=
' Y'),
 Row(ID=21, Student_ID='Null', Gender='F', AGE=17, Score=6.8, CLASS='
Y '),
 Row(ID=22, Student_ID='12744', Gender='F', AGE=14, Score=7.1, CLASS=
'  y'),
 Row(ID=23, Student_ID='34302', Gender='F', AGE=15, Score=6.5, CLASS=
'y '),
 Row(ID=24, Student_ID=None, Gender='M', AGE=16, Score=5.5, CLASS='
Y'),
 Row(ID=25, Student_ID='34242', Gender='F', AGE=17, Score=6.8, CLASS=
'y '),
 Row(ID=26, Student_ID='46675', Gender='F', AGE=15, Score=6.7, CLASS=
'y '),
 Row(ID=27, Student_ID='45566', Gender='M', AGE=16, Score=6.5, CLASS=
'y '),
 Row(ID=28, Student_ID='34309', Gender='M', AGE=17, Score=5.5, CLASS=
'y '),
 Row(ID=29, Student_ID='87664', Gender='M', AGE=14, Score=6.8, CLASS=
' Y'),
 Row(ID=30, Student_ID='34245', Gender='F', AGE=15, Score=7.1, CLASS=
'y ')]
```

In [36]:
```
# Get basic statistics
summary = df.describe()
summary.show()
```

```
+-------+-----------------+-----------------+------+--------------
---+-----------------+-----+
|summary|               ID|       Student_ID|Gender|
AGE|            Score|CLASS|
+-------+-----------------+-----------------+------+--------------
---+-----------------+-----+
|  count|               29|               29|    30|
30|               30|   30|
|   mean|15.241379310344827| 41860.82142857143|  null|15.566666666666
666|             5.73| null|
| stddev| 9.276141332987368|20171.267078682085|  null| 1.072648457158
112|1.5783339098665028| null|
|    min|                1|            12744|     F|
14|              2.0|    Y|
|    max|               30|             Null|     M|
17|              7.1|   y |
+-------+-----------------+-----------------+------+--------------
---+-----------------+-----+
```

In [37]:
```
# Get data types and missing values
info = df.printSchema()
info
```

```
root
 |-- ID: integer (nullable = true)
 |-- Student_ID: string (nullable = true)
 |-- Gender: string (nullable = true)
 |-- AGE: integer (nullable = true)
 |-- Score: double (nullable = true)
 |-- CLASS: string (nullable = true)
```

In [38]:
```
# Find missing values in the specified column
missing_values = df.filter(col("ID").isNull() | isnan(col("ID")))
missing_values.show()
```

```
+----+----------+------+---+-----+-----+
|  ID|Student_ID|Gender|AGE|Score|CLASS|
+----+----------+------+---+-----+-----+
|null|     87656|     F| 14|  6.8|   y |
+----+----------+------+---+-----+-----+
```

In [39]:
```
null_count = df.filter(col("Student_ID").isNull()).count()
null_count
```

Out[39]: 1

In [40]:
```
# Check for duplicate rows
duplicates = df.groupBy(df.ID).count().filter("count > 1")
duplicates.show()
```

```
+---+-----+
| ID|count|
+---+-----+
|  2|    2|
+---+-----+
```

In [31]:
```
# Print results
summary.show()
info
missing_values.show()
duplicates.show()
```

```
+-------+-----------------+-----------------+------+--------------
---+-----------------+-----+
|summary|               ID|       Student_ID|Gender|
AGE|            Score|CLASS|
+-------+-----------------+-----------------+------+--------------
---+-----------------+-----+
|  count|               30|               29|    30|
30|               30|   30|
|   mean|15.133333333333333| 41860.82142857143|  null|15.566666666666
666|             5.73| null|
| stddev| 9.133996116567559|20171.267078682085|  null| 1.072648457158
112|1.5783339098665028| null|
|    min|                1|            12744|     F|
14|              2.0|    Y|
|    max|               30|             Null|     M|
17|              7.1|   y |
+-------+-----------------+-----------------+------+--------------
---+-----------------+-----+

+---+----------+------+---+-----+-----+
| ID|Student_ID|Gender|AGE|Score|CLASS|
+---+----------+------+---+-----+-----+
+---+----------+------+---+-----+-----+

+---+-----+
| ID|count|
+---+-----+
|  2|    2|
+---+-----+
```