DSC 650

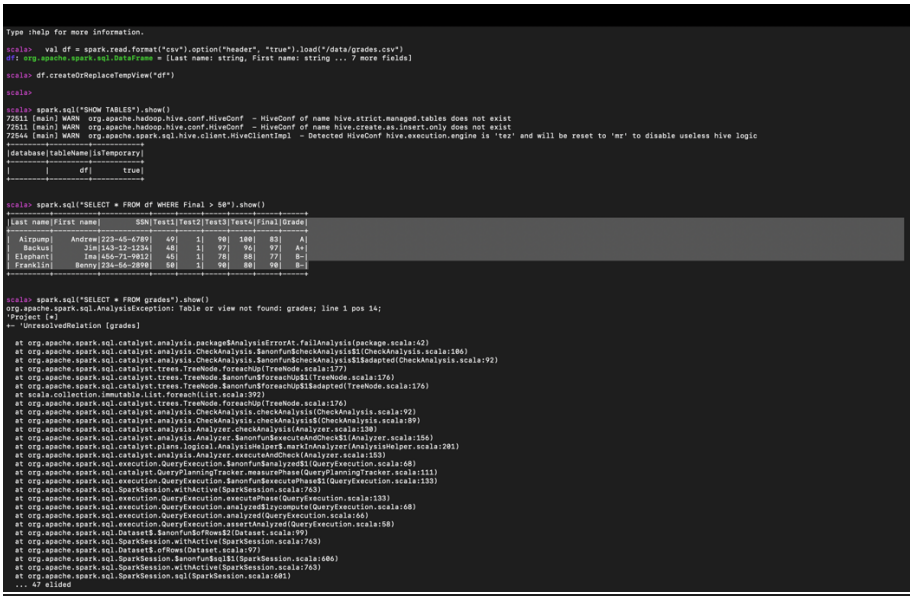
Inman, Gracie

Week 5

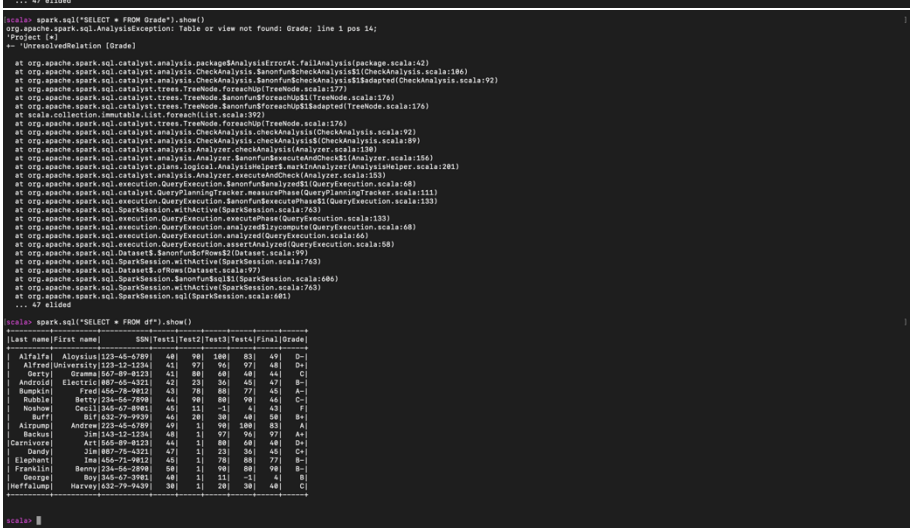
04/14/24

Spark SQL

- Results of SparkSQL



```
Type :help for more information.

scala>   val df = spark.read.format("csv").option("header", "true").load("/data/grades.csv")
df: org.apache.spark.sql.DataFrame = [Last name: string, First name: string ... 7 more fields]

scala> df.createOrReplaceTempView("df")

scala>

scala> spark.sql("SHOW TABLES").show()
72511 [main] WARN  org.apache.hadoop.hive.conf.HiveConf  - HiveConf of name hive.strict.managed.tables does not exist
72511 [main] WARN  org.apache.hadoop.hive.conf.HiveConf  - HiveConf of name hive.create.as.insert.only does not exist
72544 [main] WARN  org.apache.spark.sql.hive.client.HiveClientImpl  - Detected HiveConf hive.execution.engine is 'tez' and will be reset to 'mr' to disable useless hive logic
+--------+---------+-----------+
|database|tableName|isTemporary|
+--------+---------+-----------+
|        |       df|       true|
+--------+---------+-----------+


scala> spark.sql("SELECT * FROM df WHERE Final > 50").show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|543-12-123A|   48|    1|   97|   96|   97|   A+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+


scala> spark.sql("SELECT * FROM grades").show()
org.apache.spark.sql.AnalysisException: Table or view not found: grades; line 1 pos 14;
'Project [*]
+- 'UnresolvedRelation [grades]

  at org.apache.spark.sql.catalyst.analysis.package$AnalysisErrorAt.failAnalysis(package.scala:42)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.$anonfun$checkAnalysis$1(CheckAnalysis.scala:106)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.$anonfun$checkAnalysis$1$adapted(CheckAnalysis.scala:92)
  at org.apache.spark.sql.catalyst.trees.TreeNode.foreachUp(TreeNode.scala:177)
  at org.apache.spark.sql.catalyst.trees.TreeNode.$anonfun$foreachUp$1(TreeNode.scala:176)
  at org.apache.spark.sql.catalyst.trees.TreeNode.$anonfun$foreachUp$1$adapted(TreeNode.scala:176)
  at scala.collection.immutable.List.foreach(List.scala:392)
  at org.apache.spark.sql.catalyst.trees.TreeNode.foreachUp(TreeNode.scala:176)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.checkAnalysis(CheckAnalysis.scala:92)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.checkAnalysis$(CheckAnalysis.scala:89)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.checkAnalysis(Analyzer.scala:130)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.$anonfun$executeAndCheck$1(Analyzer.scala:156)
  at org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper$.markInAnalyzer(AnalysisHelper.scala:201)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.executeAndCheck(Analyzer.scala:153)
  at org.apache.spark.sql.execution.QueryExecution.$anonfun$analyzed$1(QueryExecution.scala:68)
  at org.apache.spark.sql.catalyst.QueryPlanningTracker.measurePhase(QueryPlanningTracker.scala:111)
  at org.apache.spark.sql.execution.QueryExecution.$anonfun$executePhase$1(QueryExecution.scala:133)
  at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:763)
  at org.apache.spark.sql.execution.QueryExecution.executePhase(QueryExecution.scala:133)
  at org.apache.spark.sql.execution.QueryExecution.analyzed$lzycompute(QueryExecution.scala:68)
  at org.apache.spark.sql.execution.QueryExecution.analyzed(QueryExecution.scala:66)
  at org.apache.spark.sql.execution.QueryExecution.assertAnalyzed(QueryExecution.scala:58)
  at org.apache.spark.sql.Dataset$.$anonfun$ofRows$2(Dataset.scala:99)
  at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:763)
  at org.apache.spark.sql.Dataset$.ofRows(Dataset.scala:97)
  at org.apache.spark.sql.SparkSession.$anonfun$sql$1(SparkSession.scala:606)
  at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:763)
  at org.apache.spark.sql.SparkSession.sql(SparkSession.scala:601)
  ... 47 elided
```



```
scala> spark.sql("SELECT * FROM Grade").show()
org.apache.spark.sql.AnalysisException: Table or view not found: Grade; line 1 pos 14;
'Project [*]
+- 'UnresolvedRelation [Grade]

  at org.apache.spark.sql.catalyst.analysis.package$AnalysisErrorAt.failAnalysis(package.scala:42)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.$anonfun$checkAnalysis$1(CheckAnalysis.scala:106)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.$anonfun$checkAnalysis$1$adapted(CheckAnalysis.scala:92)
  at org.apache.spark.sql.catalyst.trees.TreeNode.foreachUp(TreeNode.scala:177)
  at org.apache.spark.sql.catalyst.trees.TreeNode.$anonfun$foreachUp$1(TreeNode.scala:176)
  at org.apache.spark.sql.catalyst.trees.TreeNode.$anonfun$foreachUp$1$adapted(TreeNode.scala:176)
  at scala.collection.immutable.List.foreach(List.scala:392)
  at org.apache.spark.sql.catalyst.trees.TreeNode.foreachUp(TreeNode.scala:176)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.checkAnalysis(CheckAnalysis.scala:92)
  at org.apache.spark.sql.catalyst.analysis.CheckAnalysis.checkAnalysis$(CheckAnalysis.scala:89)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.checkAnalysis(Analyzer.scala:130)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.$anonfun$executeAndCheck$1(Analyzer.scala:156)
  at org.apache.spark.sql.catalyst.plans.logical.AnalysisHelper$.markInAnalyzer(AnalysisHelper.scala:201)
  at org.apache.spark.sql.catalyst.analysis.Analyzer.executeAndCheck(Analyzer.scala:153)
  at org.apache.spark.sql.execution.QueryExecution.$anonfun$analyzed$1(QueryExecution.scala:68)
  at org.apache.spark.sql.catalyst.QueryPlanningTracker.measurePhase(QueryPlanningTracker.scala:111)
  at org.apache.spark.sql.execution.QueryExecution.$anonfun$executePhase$1(QueryExecution.scala:133)
  at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:763)
  at org.apache.spark.sql.execution.QueryExecution.executePhase(QueryExecution.scala:133)
  at org.apache.spark.sql.execution.QueryExecution.analyzed$lzycompute(QueryExecution.scala:68)
  at org.apache.spark.sql.execution.QueryExecution.analyzed(QueryExecution.scala:66)
  at org.apache.spark.sql.execution.QueryExecution.assertAnalyzed(QueryExecution.scala:58)
  at org.apache.spark.sql.Dataset$.$anonfun$ofRows$2(Dataset.scala:99)
  at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:763)
  at org.apache.spark.sql.Dataset$.ofRows(Dataset.scala:97)
  at org.apache.spark.sql.SparkSession.$anonfun$sql$1(SparkSession.scala:606)
  at org.apache.spark.sql.SparkSession.withActive(SparkSession.scala:763)
  at org.apache.spark.sql.SparkSession.sql(SparkSession.scala:601)
  ... 47 elided

scala> spark.sql("SELECT * FROM df").show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Alfalfa|   Aloysius|123-45-6789|   40|   98|  100|   83|   49|   D-|
|    Alfred|University|123-12-1234|   41|   97|   96|   97|   48|   D+|
|    Gerty|    Gramma|567-89-0123|   41|   80|   60|   40|   44|    C|
|  Android|  Electric|087-65-4321|   42|   23|   36|   45|   47|   B-|
|  Bumpkin|       Fred|656-78-9012|   43|   78|   88|   77|   45|   A-|
|    Rubble|     Betty|234-56-7890|   44|   90|   80|   90|   46|   C-|
|    Noshow|     Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|
|      Buff|       Bif|632-79-9939|   46|   20|   30|   40|   50|   B+|
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|143-12-123A|   48|    1|   97|   96|   97|   A+|
|Carnivore|       Art|565-89-0123|   44|    1|   88|   60|   48|   D+|
|    Dandy|       Jim|087-75-4321|   47|    1|   23|   36|   45|   C+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
|   George|       Boy|345-67-3901|   40|    1|   11|   -1|    4|    B|
|Heffalump|    Harvey|632-79-9439|   30|    1|   20|   30|   40|    C|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+

scala>
```

- Three additional Queries
    - Calculate each student's average test scores and display in a new column.

```
scala> val averageScoresDF = df.withColumn("AverageScore", ($"Test1" + $"Test2" + $"Test3" + $"Test4" + $"Final") / 5)
averageScoresDF: org.apache.spark.sql.DataFrame = [Last name: string, First name: string ... 8 more fields]

scala> averageScoresDF.show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+------------+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|AverageScore|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+------------+
|  Alfalfa|  Aloysius|123-45-6789|   40|   90|  100|   83|   49|   D-|        72.4|
|   Alfred|University|123-12-1234|   41|   97|   96|   97|   48|   D+|        75.8|
|     Gerty|    Gramma|567-89-0123|   41|   80|   60|   40|   44|    C|        53.0|
|  Android|  Electric|087-65-4321|   42|   23|   36|   45|   47|   B-|        38.6|
|  Bumpkin|      Fred|456-78-9012|   43|   78|   88|   77|   45|   A-|        66.2|
|   Rubble|     Betty|234-56-7890|   44|   90|   80|   90|   46|   C-|        70.0|
|   Noshow|     Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|        20.4|
|     Buff|       Bif|632-79-9939|   46|   20|   30|   40|   50|   B+|        37.2|
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|        64.6|
|   Backus|       Jim|143-12-1234|   48|    1|   97|   96|   97|   A+|        67.8|
|Carnivore|       Art|565-89-0123|   44|    1|   80|   60|   40|   D+|        45.0|
|     Dandy|       Jim|087-75-4321|   47|    1|   23|   36|   45|   C+|        30.4|
| Elephant|       Ima|456-71-9812|   45|    1|   78|   88|   77|   B-|        57.8|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|        62.2|
|   George|       Boy|345-67-3901|   40|    1|   11|   -1|    4|    B|        11.0|
|Heffalump|    Harvey|632-79-9439|   30|    1|   20|   30|   40|    C|        24.2|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+------------+
```

    - Calculate the max score on each test.

```
scala> val maxScoresDF = df.select(max($"Test1").alias("MaxTest1"), max($"Test2").alias("MaxTest2"), max($"Test3").alias("MaxTest3"), max($"Test4").alias("MaxTest4"), max($"Final").alias("MaxFinal"))
maxScoresDF: org.apache.spark.sql.DataFrame = [MaxTest1: string, MaxTest2: string ... 3 more fields]

scala> maxScoresDF.show()
+--------+--------+--------+--------+--------+
|MaxTest1|MaxTest2|MaxTest3|MaxTest4|MaxFinal|
+--------+--------+--------+--------+--------+
|      50|      97|      97|      97|      97|
+--------+--------+--------+--------+--------+


scala>
```

    - Select for grades D and F

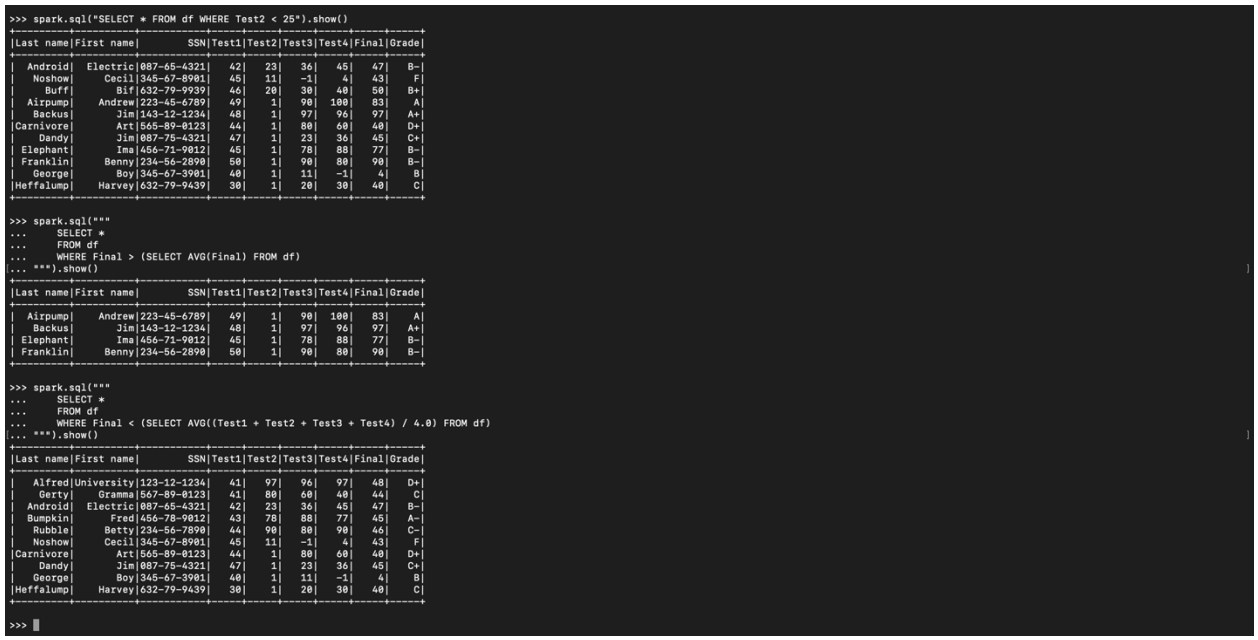
```
scala> val filteredGradesDF = df.filter(col("Grade").like("%D%") || col("Grade").like("%F%"))
filteredGradesDF: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Last name: string, First name: string ... 7 more fields]

scala> filteredGradesDF.show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Alfalfa|  Aloysius|123-45-6789|   40|   90|  100|   83|   49|   D-|
|   Alfred|University|123-12-1234|   41|   97|   96|   97|   48|   D+|
|   Noshow|     Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|
|Carnivore|       Art|565-89-0123|   44|    1|   80|   60|   40|   D+|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
```

- PySpark
    - Specified Commands

```
+- 'UnresolvedRelation [grades]

>>>     df = spark.read.format('csv').option('header', 'true').load('/data/grades.csv')
  File "<stdin>", line 1
    df = spark.read.format('csv').option('header', 'true').load('/data/grades.csv')
    ^
IndentationError: unexpected indent
>>> df = spark.read.format('csv').option('header', 'true').load('/data/grades.csv')
>>> df.show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Alfalfa|  Aloysius|123-45-6789|   40|   90|  100|   83|   49|   D-|
|   Alfred|University|123-12-1234|   41|   97|   96|   97|   48|   D+|
|    Gerty|    Gramma|567-89-0123|   41|   80|   60|   40|   44|    C|
|  Android|  Electric|087-65-4321|   42|   23|   36|   45|   47|   B-|
|  Bumpkin|      Fred|456-78-9012|   43|   78|   88|   77|   45|   A-|
|   Rubble|     Betty|234-56-7890|   44|   90|   80|   90|   46|   C-|
|   Noshow|     Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|
|     Buff|       Bif|632-79-9939|   46|   20|   30|   40|   50|   B+|
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|143-12-1234|   48|    1|   97|   96|   97|   A+|
|Carnivore|       Art|565-89-0123|   44|    1|   80|   60|   40|   D+|
|    Dandy|       Jim|087-75-4321|   47|    1|   23|   36|   45|   C+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
|   George|       Boy|345-67-3901|   40|    1|   11|   -1|    4|    B|
|Heffalump|    Harvey|632-79-9439|   30|    1|   20|   30|   40|    C|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+

>>> df.createOrReplaceTempView('df')
>>> spark.sql('SHOW TABLES').show()
+--------+---------+-----------+
|database|tableName|isTemporary|
+--------+---------+-----------+
|        |       df|       true|
+--------+---------+-----------+

>>> spark.sql('SELECT * FROM df WHERE Final > 50').show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|143-12-1234|   48|    1|   97|   96|   97|   A+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
```

```
>>> spark.sql('SELECT * FROM df').show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Alfalfa|  Aloysius|123-45-6789|   40|   90|  100|   83|   49|   D-|
|   Alfred|University|123-12-1234|   41|   97|   96|   97|   48|   D+|
|    Gerty|    Gramma|567-89-0123|   41|   80|   60|   40|   44|    C|
|  Android|  Electric|087-65-4321|   42|   23|   36|   45|   47|   B-|
|  Bumpkin|      Fred|456-78-9012|   43|   78|   88|   77|   45|   A-|
|   Rubble|     Betty|234-56-7890|   44|   90|   80|   90|   46|   C-|
|   Noshow|     Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|
|     Buff|       Bif|632-79-9939|   46|   20|   30|   40|   50|   B+|
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|143-12-1234|   48|    1|   97|   96|   97|   A+|
|Carnivore|       Art|565-89-0123|   44|    1|   80|   60|   40|   D+|
|    Dandy|       Jim|087-75-4321|   47|    1|   23|   36|   45|   C+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
|   George|       Boy|345-67-3901|   40|    1|   11|   -1|    4|    B|
|Heffalump|    Harvey|632-79-9439|   30|    1|   20|   30|   40|    C|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
```

- Three additional commands
  - Filtered to show everyone who got less than a 25 on Test 2
  - Filtered to show those who got above average on the final
  - Filtered to show those whose average score on tests 1-4 were lower than their score on the final



```
>>> spark.sql("SELECT * FROM df WHERE Test2 < 25").show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Android|  Electric|087-65-4321|   42|   23|   36|   45|   47|   B-|
|   Noshow|      Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|
|     Buff|       Bif|632-79-9939|   46|   20|   30|   40|   50|   B+|
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|143-12-1234|   48|    1|   97|   96|   97|   A+|
|Carnivore|       Art|565-89-0123|   44|    1|   80|   60|   40|   D+|
|     Dandy|       Jim|087-75-4321|   47|    1|   23|   36|   45|   C+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
|   George|       Boy|345-67-3901|   40|    1|   11|   -1|    4|    B|
|Heffalump|    Harvey|632-79-9439|   30|    1|   20|   30|   40|    C|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+

>>> spark.sql("""
...     SELECT *
...     FROM df
...     WHERE Final > (SELECT AVG(Final) FROM df)
... """).show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|  Airpump|    Andrew|223-45-6789|   49|    1|   90|  100|   83|    A|
|   Backus|       Jim|143-12-1234|   48|    1|   97|   96|   97|   A+|
| Elephant|       Ima|456-71-9012|   45|    1|   78|   88|   77|   B-|
| Franklin|     Benny|234-56-2890|   50|    1|   90|   80|   90|   B-|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+

>>> spark.sql("""
...     SELECT *
...     FROM df
...     WHERE Final < (SELECT AVG((Test1 + Test2 + Test3 + Test4) / 4.0) FROM df)
... """).show()
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|Last name|First name|        SSN|Test1|Test2|Test3|Test4|Final|Grade|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+
|   Alfred|University|123-12-1234|   41|   97|   96|   97|   48|   D+|
|     Gerty|     Gramma|567-89-0123|   41|   80|   60|   40|   44|    C|
|  Android|  Electric|087-65-4321|   42|   23|   36|   45|   47|   B-|
|  Bumpkin|       Fred|456-78-9012|   43|   78|   88|   77|   45|   A-|
|   Rubble|     Betty|234-56-7890|   44|   90|   80|   90|   46|   C-|
|   Noshow|      Cecil|345-67-8901|   45|   11|   -1|    4|   43|    F|
|Carnivore|       Art|565-89-0123|   44|    1|   80|   60|   40|   D+|
|     Dandy|       Jim|087-75-4321|   47|    1|   23|   36|   45|   C+|
|   George|       Boy|345-67-3901|   40|    1|   11|   -1|    4|    B|
|Heffalump|    Harvey|632-79-9439|   30|    1|   20|   30|   40|    C|
+---------+----------+-----------+-----+-----+-----+-----+-----+-----+

>>>
```

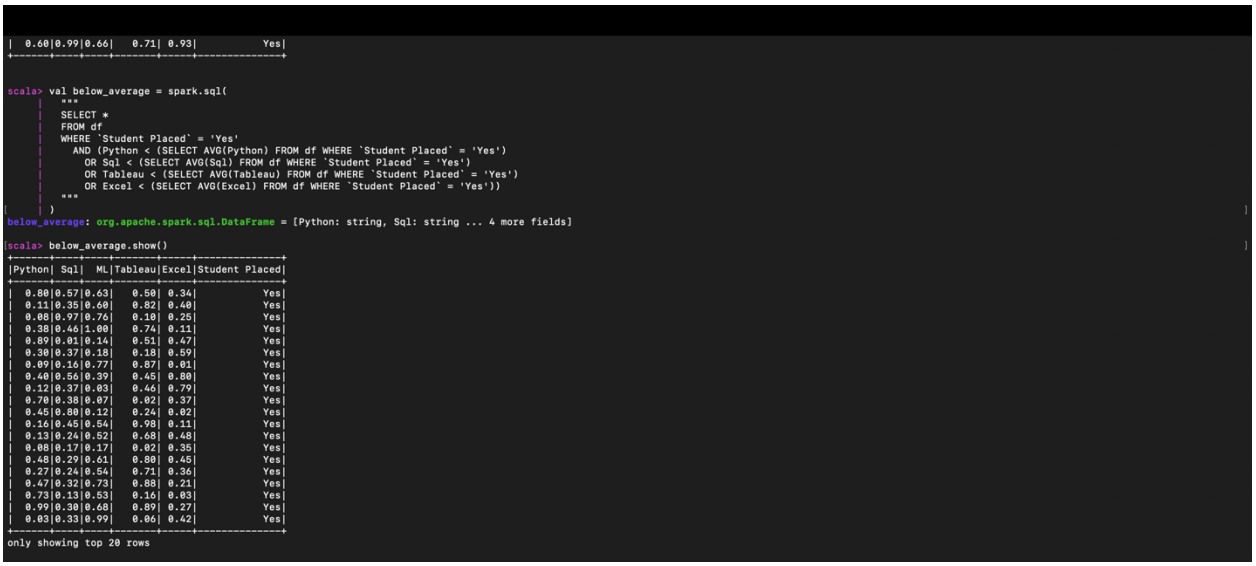
- Practice with Scores Dataset
    - Load Data

```
      /_/

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 1.8.0_275)
Type in expressions to have them evaluated.
Type :help for more information.

scala>    val df = spark.read.format("csv").option("header", "true").load("/data/scores.csv")
df: org.apache.spark.sql.DataFrame = [Python: string, Sql: string ... 4 more fields]

scala> df.createOrReplaceTempView("df")

scala> spark.sql("SHOW TABLES").show()
202908 [main] WARN  org.apache.hadoop.hive.conf.HiveConf  - HiveConf of name hive.strict.managed.tables does not exist
202909 [main] WARN  org.apache.hadoop.hive.conf.HiveConf  - HiveConf of name hive.create.as.insert.only does not exist
202942 [main] WARN  org.apache.spark.sql.hive.client.HiveClientImpl  - Detected HiveConf hive.execution.engine is 'tez' and will be reset to 'mr' to disable useless hive logic
+--------+---------+-----------+
|database|tableName|isTemporary|
+--------+---------+-----------+
|        |       df|       true|
+--------+---------+-----------+


scala> spark.sql("SELECT * FROM df").show()
+------+----+----+-------+-----+--------------+
|Python| Sql|  ML|Tableau|Excel|Student Placed|
+------+----+----+-------+-----+--------------+
|  0.80|0.57|0.63|   0.50| 0.34|           Yes|
|  0.81|0.90|0.62|   0.71| 0.92|            No|
|  0.49|0.69|0.62|   0.64| 0.41|            No|
|  0.40|0.94|0.60|   0.26| 0.47|            No|
|  0.31|0.87|1.00|   0.23| 0.99|            No|
|  0.14|0.87|0.09|   0.92| 0.70|            No|
|  0.21|0.80|0.88|   0.63| 0.36|            No|
|  0.08|0.78|0.61|   0.40| 0.63|            No|
|  0.81|0.17|0.90|   0.50| 0.61|            No|
|  0.31|0.28|0.79|   0.12| 0.67|            No|
|  0.28|0.13|0.46|   1.00| 0.06|            No|
|  0.22|0.63|0.50|   0.02| 0.29|            No|
|  0.86|0.67|0.65|   0.31| 0.32|            No|
|  0.91|0.21|0.22|   0.74| 0.37|            No|
|  0.88|0.83|0.93|   0.19| 0.63|            No|
|  0.11|0.35|0.60|   0.82| 0.40|           Yes|
|  0.08|0.97|0.76|   0.10| 0.25|           Yes|
|  0.38|0.46|1.00|   0.74| 0.11|           Yes|
|  0.89|0.01|0.14|   0.51| 0.47|           Yes|
|  0.30|0.37|0.18|   0.18| 0.59|           Yes|
+------+----+----+-------+-----+--------------+
only showing top 20 rows
```

    - Filter to show those who scored above average in all subjects.

```
scala> val above_average = spark.sql(
     |   """
     |   SELECT *
     |   FROM df
     |   WHERE Python > (SELECT AVG(Python) FROM df)
     |     AND Sql > (SELECT AVG(Sql) FROM df)
     |     AND Tableau > (SELECT AVG(Tableau) FROM df)
     |     AND Excel > (SELECT AVG(Excel) FROM df)
     |   """
     | )
above_average: org.apache.spark.sql.DataFrame = [Python: string, Sql: string ... 4 more fields]

scala> above_average.show()
+------+----+----+-------+-----+--------------+
|Python| Sql|  ML|Tableau|Excel|Student Placed|
+------+----+----+-------+-----+--------------+
|  0.81|0.90|0.62|   0.71| 0.92|            No|
|  0.52|0.79|0.27|   0.86| 0.51|            No|
|  0.94|0.61|0.07|   0.87| 0.88|           Yes|
|  0.56|0.63|0.51|   0.86| 0.81|           Yes|
|  0.90|0.75|0.54|   0.83| 0.53|            No|
|  0.74|0.84|0.62|   0.87| 0.55|            No|
|  0.70|0.82|0.92|   0.52| 0.93|           Yes|
|  0.80|0.62|0.53|   0.57| 0.90|           Yes|
|  0.73|0.96|0.30|   0.65| 0.85|           Yes|
|  0.82|0.69|0.29|   0.62| 0.84|           Yes|
|  0.90|0.69|0.47|   0.73| 0.62|           Yes|
|  0.60|0.84|0.70|   0.82| 0.54|            No|
|  0.81|0.70|0.60|   0.55| 0.57|            No|
|  0.86|0.83|0.39|   0.93| 0.63|           Yes|
|  0.65|0.69|0.24|   0.64| 0.88|           Yes|
|  0.95|0.65|0.62|   0.74| 0.61|           Yes|
|  0.60|0.99|0.66|   0.71| 0.93|           Yes|
+------+----+----+-------+-----+--------------+
```

- Filter to show those who were placed and scored below average in one subject.



```
|  0.60|0.99|0.66|   0.71| 0.93|          Yes|
+------+----+----+-------+-----+-------------+


scala> val below_average = spark.sql(
     |   """
     |   SELECT *
     |   FROM df
     |   WHERE `Student Placed` = 'Yes'
     |     AND (Python < (SELECT AVG(Python) FROM df WHERE `Student Placed` = 'Yes')
     |       OR Sql < (SELECT AVG(Sql) FROM df WHERE `Student Placed` = 'Yes')
     |       OR Tableau < (SELECT AVG(Tableau) FROM df WHERE `Student Placed` = 'Yes')
     |       OR Excel < (SELECT AVG(Excel) FROM df WHERE `Student Placed` = 'Yes'))
     |   """
     | )
below_average: org.apache.spark.sql.DataFrame = [Python: string, Sql: string ... 4 more fields]

scala> below_average.show()
+------+----+----+-------+-----+--------------+
|Python| Sql|  ML|Tableau|Excel|Student Placed|
+------+----+----+-------+-----+--------------+
|  0.80|0.57|0.63|   0.50| 0.34|           Yes|
|  0.11|0.35|0.60|   0.82| 0.40|           Yes|
|  0.88|0.97|0.76|   0.10| 0.25|           Yes|
|  0.38|0.46|1.00|   0.74| 0.11|           Yes|
|  0.89|0.01|0.14|   0.51| 0.47|           Yes|
|  0.30|0.37|0.18|   0.18| 0.59|           Yes|
|  0.09|0.16|0.77|   0.87| 0.01|           Yes|
|  0.40|0.56|0.39|   0.45| 0.80|           Yes|
|  0.12|0.37|0.03|   0.46| 0.79|           Yes|
|  0.70|0.38|0.07|   0.02| 0.37|           Yes|
|  0.45|0.80|0.12|   0.24| 0.02|           Yes|
|  0.16|0.45|0.54|   0.98| 0.11|           Yes|
|  0.13|0.24|0.52|   0.68| 0.48|           Yes|
|  0.88|0.17|0.17|   0.02| 0.35|           Yes|
|  0.48|0.29|0.61|   0.88| 0.45|           Yes|
|  0.27|0.24|0.54|   0.71| 0.36|           Yes|
|  0.47|0.32|0.73|   0.88| 0.21|           Yes|
|  0.73|0.13|0.53|   0.16| 0.03|           Yes|
|  0.99|0.30|0.68|   0.89| 0.27|           Yes|
|  0.03|0.33|0.99|   0.06| 0.42|           Yes|
+------+----+----+-------+-----+--------------+
only showing top 20 rows
```

- Filter to show those who were not placed but scored higher than average in at least one subject.

```
scala> val above_average_not_placed = spark.sql(
     |   """
     |   SELECT *
     |   FROM df
     |   WHERE `Student Placed` = 'No'
     |     AND (Python > (SELECT AVG(Python) FROM df WHERE `Student Placed` = 'No')
     |       OR Sql > (SELECT AVG(Sql) FROM df WHERE `Student Placed` = 'No')
     |       OR Tableau > (SELECT AVG(Tableau) FROM df WHERE `Student Placed` = 'No')
     |       OR Excel > (SELECT AVG(Excel) FROM df WHERE `Student Placed` = 'No'))
     |   """
     | )
above_average_not_placed: org.apache.spark.sql.DataFrame = [Python: string, Sql: string ... 4 more fields]

scala> above_average_not_placed.show()
+------+----+----+-------+-----+--------------+
|Python| Sql|  ML|Tableau|Excel|Student Placed|
+------+----+----+-------+-----+--------------+
|  0.81|0.90|0.62|   0.71| 0.92|            No|
|  0.49|0.69|0.62|   0.64| 0.41|            No|
|  0.40|0.94|0.60|   0.26| 0.47|            No|
|  0.31|0.87|1.00|   0.23| 0.99|            No|
|  0.14|0.87|0.09|   0.92| 0.70|            No|
|  0.21|0.80|0.88|   0.63| 0.36|            No|
|  0.08|0.78|0.61|   0.40| 0.63|            No|
|  0.81|0.17|0.90|   0.50| 0.61|            No|
|  0.31|0.28|0.79|   0.12| 0.67|            No|
|  0.28|0.13|0.46|   1.00| 0.06|            No|
|  0.22|0.63|0.50|   0.02| 0.29|            No|
|  0.86|0.67|0.65|   0.31| 0.32|            No|
|  0.91|0.21|0.22|   0.74| 0.37|            No|
|  0.88|0.83|0.93|   0.19| 0.63|            No|
|  0.83|0.65|0.00|   0.40| 0.04|            No|
|  0.02|0.26|0.65|   0.56| 0.32|            No|
|  0.53|0.22|0.77|   0.72| 0.22|            No|
|  0.38|0.44|0.00|   0.42| 0.85|            No|
|  0.56|0.57|0.38|   0.31| 0.15|            No|
|  0.00|0.36|0.84|   0.75| 0.11|            No|
+------+----+----+-------+-----+--------------+
only showing top 20 rows

scala>
```