CSC 590, CSC 690 – Data Analytics
Exam #3, Fall 2024

First/Given Name: ____________________________

Last/Family Name: ____________________________

This exam contains 6 pages (including this cover page) and 5 questions.

- Clearly identify your answer for each problem, and try to organize your work in a reasonably coherent way, in the space provided. If you decided to use the back of a paper, note this clearly so the instructor can find your answer.
- Partial credit will be given for incorrect or incomplete answers that show a partial understanding of the relevant concepts. Irrelevant and meaningless answers will not receive partial credit.
- No electronic devices, including calculators, are allowed.
- Each student is allowed to use only a cheat sheet of size 8.5″×5.75″, which is equivalent to a half of a standard letter-sized paper. The cheat sheet can be used on both sides. Only handwritten cheat sheets are allowed, and each student is required to write their name on their cheat sheet. The cheat sheet must be submitted along with the exam upon completion.

Question	Points	Score
1	1.00	
2	2.00	
3	2.00	
4	2.00	
5	2.00	
Total:	9.00	

I acknowledge that it is the responsibility of every student at Missouri State University to adhere to the university's policies on Student Academic Integrity. I confirm that I have neither given nor received any unauthorized assistance during this exam.

Signature: ____________________________

1. (1.00 points) Indicate whether the following statements are true or false.

   (a) Spark provides a fault tolerant mechanism for RDDs but not for DataFrames.

   False.

   RDDs are used underneath, as a DataFrame is a RDD of Rows. Thus, the RDD lineage is still there to provide fault tolerance.

   (b) DataFrames are not immutable.

   False.

   DateFrames and RDDs cannot be modified. One can create a new one from each by applying transformations.

   (c) The following code is not *Lazy*.

   ```
   df2 = df.count()
   ```

   True.

   This is an action, and thus, it will be executed immediately.

   (d) The following code is not *Lazy*.

   ```
   df2 = df.groupBy("module").count()
   ```

   False.

   When the count() operation is applied to GroupedData (i.e., after a groupBy()), it behaves as a transformation rather than an action. Recall that transformations are lazy.

Points earned for this question: ______

2. Consider a DataFrame df whose content is as follows.

```
+----+---+----------+------+
|name|age|profession|salary|
+----+---+----------+------+
| Kia| 24|  engineer| 50000|
| Zoe| 30|   teacher| 60000|
|Amir| 28|  engineer| 70000|
| Eva| 24|accountant| 40000|
| Leo| 28|   teacher| 50000|
+----+---+----------+------+
```

(a) (1.00 points) What is the output of the following code?

```
df.selectExpr("name", "age-20 as years", "salary")\
  .filter("salary >= 50000")\
  .sort("years").show()
```

Solution:

```
+----+-----+------+
|name|years|salary|
+----+-----+------+
| Kia|    4| 50000|
|Amir|    8| 70000|
| Leo|    8| 50000|
| Zoe|   10| 60000|
+----+-----+------+
```

(b) (0.50 points) What is the output of the following code?

```
df.groupBy("profession").avg("age").show()
```

Solution:

```
+----------+--------+
|profession|avg(age)|
+----------+--------+
|   teacher|    29.0|
|  engineer|    26.0|
|accountant|    24.0|
+----------+--------+
```

(c) (0.50 points) What is the output of the following code?

```
df.orderBy(["age", "salary"], ascending=[False, True]).show()
```

Solution:

```
+----+---+----------+------+
|name|age|profession|salary|
+----+---+----------+------+
| Zoe| 30|   teacher| 60000|
| Leo| 28|   teacher| 50000|
|Amir| 28|  engineer| 70000|
| Eva| 24|accountant| 40000|
| Kia| 24|  engineer| 50000|
+----+---+----------+------+
```

Points earned for this question: ______

3. Consider two data frames with marks for different students:

```
df_590 = spark.createDataFrame([("Ari", 60), ("Joe", 70), ("Mia", 80)],
                               ["name", "CSC590"])
df_333 = spark.createDataFrame([("Ari", 80), ("Mia", 70), ("Eva", 60)],
                         ["name", "CSC311"])
```

(a) (0.50 points) What is the output of the following code? Draw the tables.

```
df_590.show()
df_333.show()
```

Solution:

```
+----+------+
|name|CSC590|
+----+------+
| Ari|    60|
| Joe|    70|
| Mia|    80|
+----+------+

+----+------+
|name|CSC311|
+----+------+
| Ari|    80|
| Mia|    70|
| Eva|    60|
+----+------+
```

(b) (0.75 points) Fill in the blanks of the following instruction so that the output of df.show() is the table shown below.

```
df = __________.join(_________, "name", "_______________")
df.show()
```

```
+----+------+------+
|name|CSC311|CSC590|
+----+------+------+
| Ari|  80  |  60  |
| Eva|  60  | NULL |
| Joe| NULL |  70  |
| Mia|  70  |  80  |
+----+------+------+
```

Solution:

```
df = df_333.join(df_590, "name", "outer")
```

(c) (0.75 points) What is the output of the following code?

```
def g(col, val):
    return sql_f.when(col.isNull(), 0).otherwise(val)

df.withColumn("x", (g(df.CSC311, df.CSC311)+g(df.CSC590, df.CSC590))/
            (g(df.CSC311, 1)+g(df.CSC590, 1))).show()
```

Solution:

```
+----+------+------+----+
|name|CSC311|CSC590|   x|
+----+------+------+----+
| Ari|    80|    60|70.0|
| Eva|    60|  NULL|60.0|
| Joe|  NULL|    70|70.0|
| Mia|    70|    80|75.0|
+----+------+------+----+
```

Points earned for this question: ______

4. Consider a data frame df constructed by the following instruction:

```
df = spark.createDataFrame([
    Row(name="Ezra", cards=["discover", "chase"], info={"age": 20, "profession": "
    student"}),
    Row(name="Reza", cards=["chase"],  info={"profession": "engineer", "age": 22}),
    Row(name="Alex", cards=["chase", "city", "discover"],  info={"age": 20}),
    Row(name="Mia", cards=[], info={"age": 22, "profession": "engineer"}),
    Row(name="Lili", cards=["citi"], info={"age": 24, "profession": "student"})
])
```

(a) (0.50 points) What is the output of the following code? Draw the tables.

```
df.select("name", sql_f.explode(df.cards).alias("card")).show()
```

Solution:

```
+----+--------+
|name|    card|
+----+--------+
|Ezra|discover|
|Ezra|   chase|
|Reza|   chase|
|Alex|   chase|
|Alex|    city|
|Alex|discover|
|Lili|    citi|
+----+--------+
```

(b) (0.75 points) What is the output of the following code? Draw the tables.

```
df2 = df.select("name", sql_f.explode(df.info).alias("key", "value")).show()
```

Solution:

```
+----+----------+--------+
|name|       key|   value|
+----+----------+--------+
|Ezra|profession| student|
|Ezra|       age|      20|
|Reza|profession|engineer|
|Reza|       age|      22|
|Alex|       age|      20|
| Mia|profession|engineer|
| Mia|       age|      22|
|Lili|profession| student|
|Lili|       age|      24|
+----+----------+--------+
```

(c) (0.75 points) What is the output of the following code?

```
df2.select("name", "value").filter("key='profession'").groupBy("name").pivot("
value").count().show()
```

Solution:

```
+----+--------+-------+
|name|engineer|student|
+----+--------+-------+
|Reza|       1|   NULL|
| Mia|       1|   NULL|
|Ezra|    NULL|      1|
|Lili|    NULL|      1|
+----+--------+-------+
```

Points earned for this question: ______

5. Consider a DataFrame df as follows.

```
+----+---+-----+----------+
|name|GPA|class|department|
+----+---+-----+----------+
| Ami|3.0| 2024|       MTH|
| Eva|3.4| 2026|       MTH|
|Alex|3.4| 2024|      COMP|
|Lisa|3.6| 2026|      COMP|
| Eli|2.9| 2026|       MTH|
|Kate|3.5| 2027|      COMP|
|Nova|3.0| 2024|      COMP|
|Levi|3.6| 2024|       MTH|
+----+---+-----+----------+
```

(a) (1.00 points) Write code that computes the average GPA of students for each department and class. The output should look like this:

```
+----------+-----+--------+
|department|class|avg(GPA)|
+----------+-----+--------+
|      COMP| 2026|     3.6|
|       MTH| 2026|    3.15|
|      COMP| 2024|     3.2|
|       MTH| 2024|     3.3|
|      COMP| 2027|     3.5|
+----------+-----+--------+
```

Solution:

```
df.groupBy("department", "class").avg("GPA").show()
```

(b) (1.00 points) Write code that computes a DataFrame that contains all students in each class whose GPAs are less than the average GPA of students in that class. The output of your code should look like the following.

```
+-----+------------------+----+---+----------+
|class|           avg_gpa|name|GPA|department|
+-----+------------------+----+---+----------+
| 2024|              3.25| Ami|3.0|       MTH|
| 2026|3.3000000000000003| Eli|2.9|       MTH|
| 2024|              3.25|Nova|3.0|      COMP|
+-----+------------------+----+---+----------+
```

Solution:

```
df2 = df.groupBy("class").agg(sql_f.avg("GPA").alias("avg_gpa"))
df3 = df2.join(df, "class").where("GPA < avg_gpa")
df3.show()
```

Points earned for this question: ______