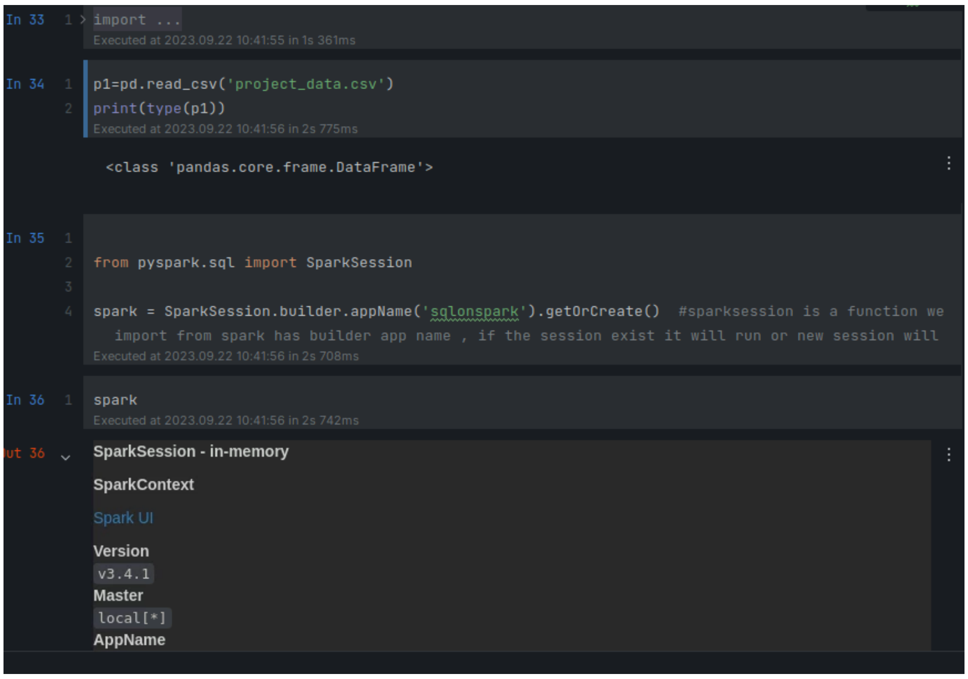
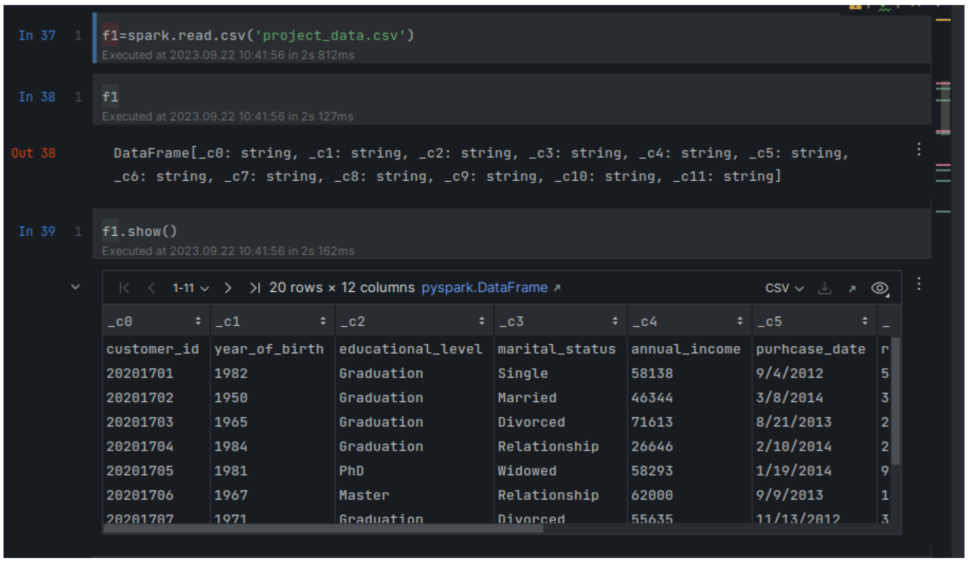
1.



```
In 33   import ...
        Executed at 2023.09.22 10:41:55 in 1s 361ms

In 34   p1=pd.read_csv('project_data.csv')
        print(type(p1))
        Executed at 2023.09.22 10:41:56 in 2s 775ms

         <class 'pandas.core.frame.DataFrame'>

In 35
        from pyspark.sql import SparkSession

        spark = SparkSession.builder.appName('sqlonspark').getOrCreate()  #sparksession is a function we
          import from spark has builder app name , if the session exist it will run or new session will
        Executed at 2023.09.22 10:41:56 in 2s 708ms

In 36   spark
        Executed at 2023.09.22 10:41:56 in 2s 742ms

Out 36  SparkSession - in-memory
        SparkContext
        Spark UI
        Version
        v3.4.1
        Master
        local[*]
        AppName
```

2.

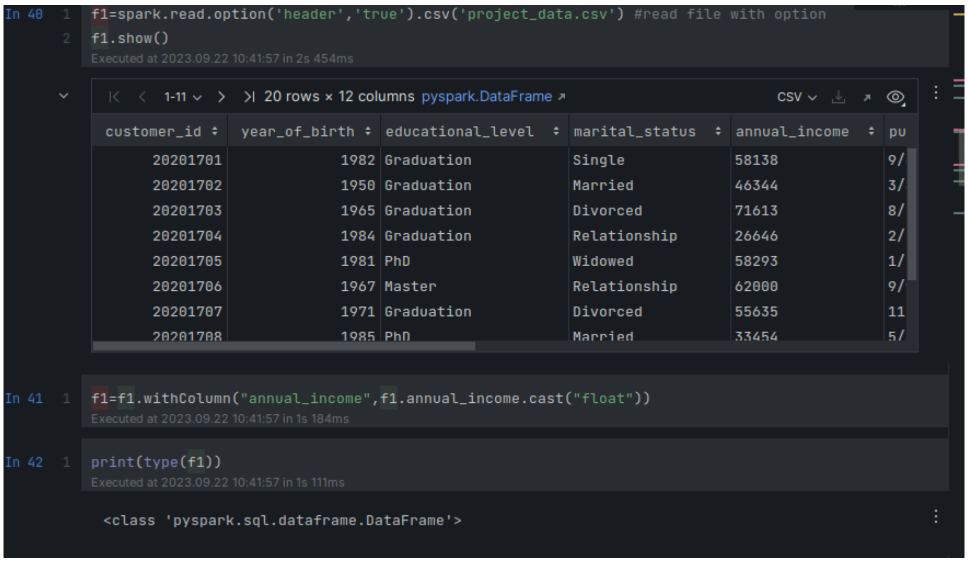


```
In 37   f1=spark.read.csv('project_data.csv')
        Executed at 2023.09.22 10:41:56 in 2s 812ms

In 38   f1
        Executed at 2023.09.22 10:41:56 in 2s 127ms

Out 38   DataFrame[_c0: string, _c1: string, _c2: string, _c3: string, _c4: string, _c5: string,
         _c6: string, _c7: string, _c8: string, _c9: string, _c10: string, _c11: string]

In 39   f1.show()
        Executed at 2023.09.22 10:41:56 in 2s 162ms
```

1-11 20 rows × 12 columns pyspark.DataFrame CSV

_c0	_c1	_c2	_c3	_c4	_c5	_
customer_id	year_of_birth	educational_level	marital_status	annual_income	purhcase_date	r
20201701	1982	Graduation	Single	58138	9/4/2012	5
20201702	1950	Graduation	Married	46344	3/8/2014	3
20201703	1965	Graduation	Divorced	71613	8/21/2013	2
20201704	1984	Graduation	Relationship	26646	2/10/2014	2
20201705	1981	PhD	Widowed	58293	1/19/2014	9
20201706	1967	Master	Relationship	62000	9/9/2013	1
20201707	1971	Graduation	Divorced	55635	11/13/2012	3

3.



In 40
```
f1=spark.read.option('header','true').csv('project_data.csv') #read file with option
f1.show()
```
Executed at 2023.09.22 10:41:57 in 2s 454ms

1-11 20 rows × 12 columns pyspark.DataFrame CSV

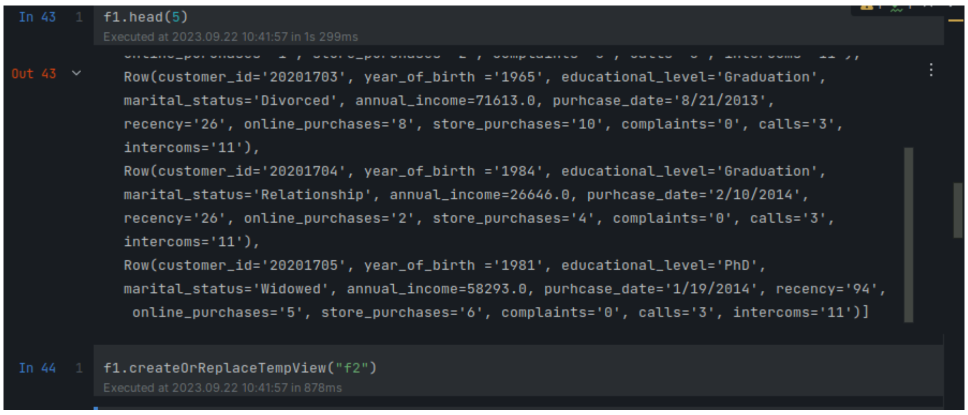
customer_id	year_of_birth	educational_level	marital_status	annual_income	pu
20201701	1982	Graduation	Single	58138	9/
20201702	1950	Graduation	Married	46344	3/
20201703	1965	Graduation	Divorced	71613	8/
20201704	1984	Graduation	Relationship	26646	2/
20201705	1981	PhD	Widowed	58293	1/
20201706	1967	Master	Relationship	62000	9/
20201707	1971	Graduation	Divorced	55635	11
20201708	1985	PhD	Married	33454	5/

In 41
```
f1=f1.withColumn("annual_income",f1.annual_income.cast("float"))
```
Executed at 2023.09.22 10:41:57 in 1s 184ms

In 42
```
print(type(f1))
```
Executed at 2023.09.22 10:41:57 in 1s 111ms

```
<class 'pyspark.sql.dataframe.DataFrame'>
```

4.

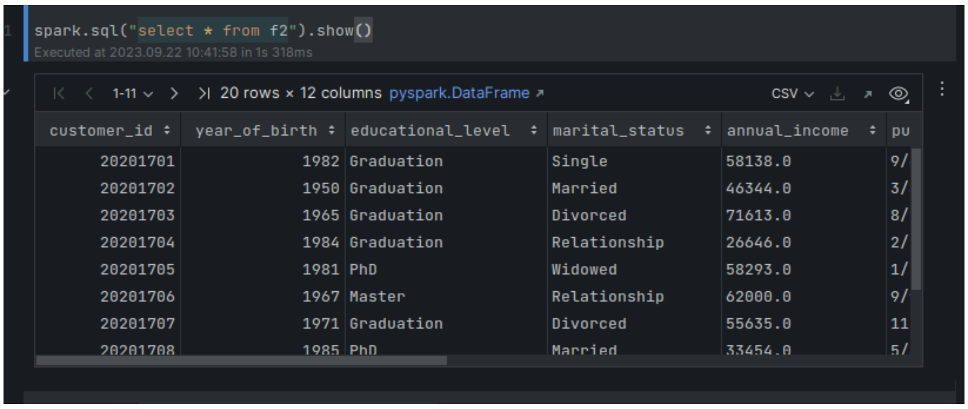


In 43
```
f1.head(5)
```
Executed at 2023.09.22 10:41:57 in 1s 299ms

Out 43
```
Row(customer_id='20201703', year_of_birth ='1965', educational_level='Graduation',
marital_status='Divorced', annual_income=71613.0, purhcase_date='8/21/2013',
recency='26', online_purchases='8', store_purchases='10', complaints='0', calls='3',
intercoms='11'),
Row(customer_id='20201704', year_of_birth ='1984', educational_level='Graduation',
marital_status='Relationship', annual_income=26646.0, purhcase_date='2/10/2014',
recency='26', online_purchases='2', store_purchases='4', complaints='0', calls='3',
intercoms='11'),
Row(customer_id='20201705', year_of_birth ='1981', educational_level='PhD',
marital_status='Widowed', annual_income=58293.0, purhcase_date='1/19/2014', recency='94',
 online_purchases='5', store_purchases='6', complaints='0', calls='3', intercoms='11')]
```

In 44
```
f1.createOrReplaceTempView("f2")
```
Executed at 2023.09.22 10:41:57 in 878ms

5.

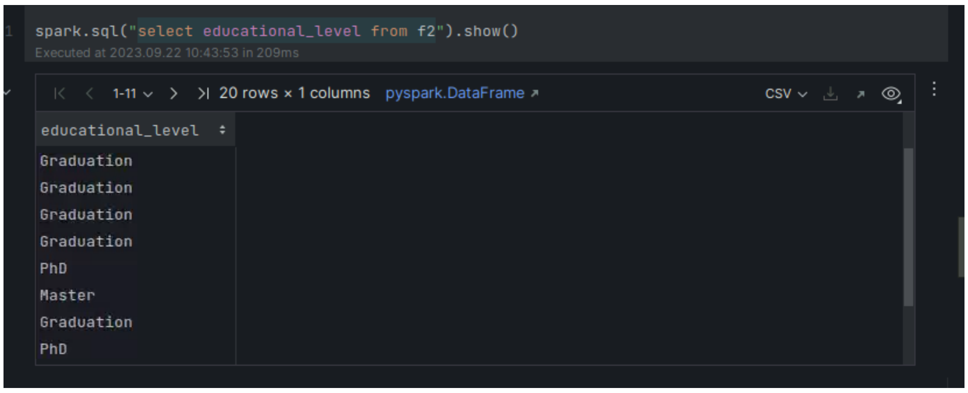


```
spark.sql("select * from f2").show()
```
Executed at 2023.09.22 10:41:58 in 1s 318ms

1-11 20 rows × 12 columns pyspark.DataFrame CSV

customer_id	year_of_birth	educational_level	marital_status	annual_income	pu
20201701	1982	Graduation	Single	58138.0	9/
20201702	1950	Graduation	Married	46344.0	3/
20201703	1965	Graduation	Divorced	71613.0	8/
20201704	1984	Graduation	Relationship	26646.0	2/
20201705	1981	PhD	Widowed	58293.0	1/
20201706	1967	Master	Relationship	62000.0	9/
20201707	1971	Graduation	Divorced	55635.0	11
20201708	1985	PhD	Married	33454.0	5/

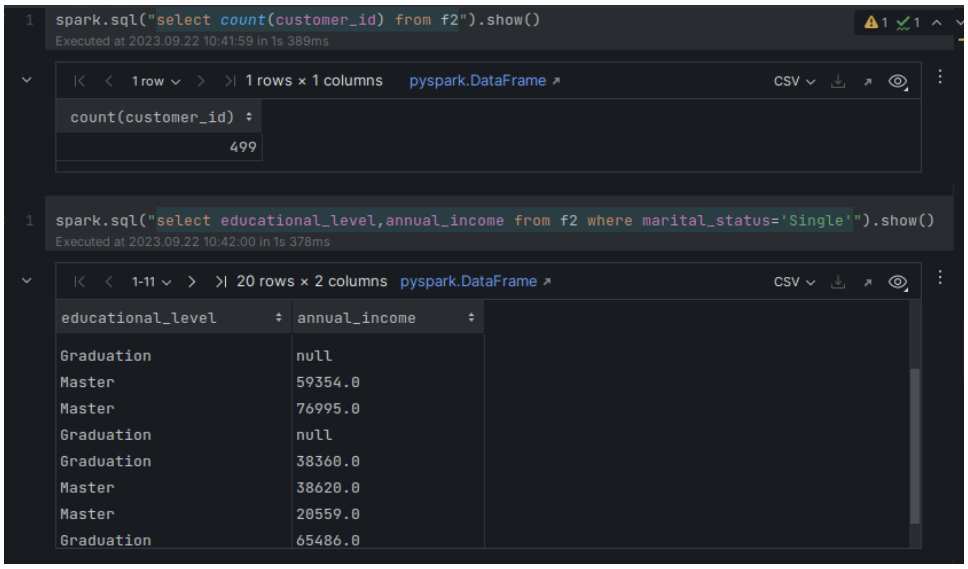
6.



```
spark.sql("select educational_level from f2").show()
Executed at 2023.09.22 10:43:53 in 209ms
```

```
1-11   20 rows × 1 columns   pyspark.DataFrame   CSV

educational_level
Graduation
Graduation
Graduation
Graduation
PhD
Master
Graduation
PhD
```

7.



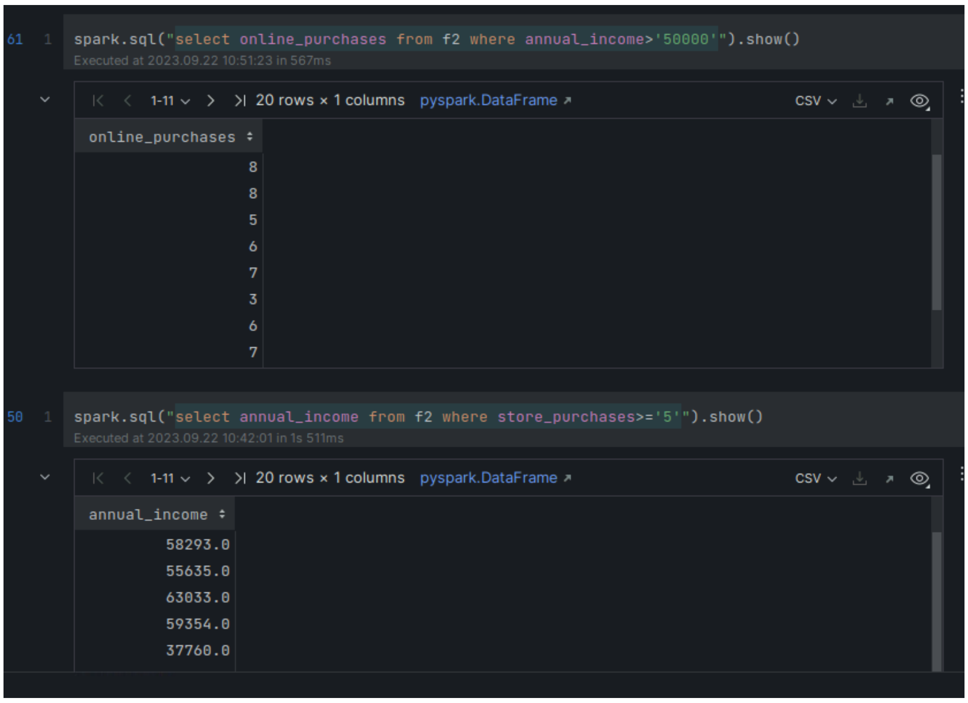
```
spark.sql("select count(customer_id) from f2").show()
Executed at 2023.09.22 10:41:59 in 1s 389ms
```

```
1 row   1 rows × 1 columns   pyspark.DataFrame   CSV

count(customer_id)
               499
```

```
spark.sql("select educational_level,annual_income from f2 where marital_status='Single'").show()
Executed at 2023.09.22 10:42:00 in 1s 378ms
```

```
1-11   20 rows × 2 columns   pyspark.DataFrame   CSV

educational_level        annual_income
Graduation               null
Master                   59354.0
Master                   76995.0
Graduation               null
Graduation               38360.0
Master                   38620.0
Master                   20559.0
Graduation               65486.0
```

8.



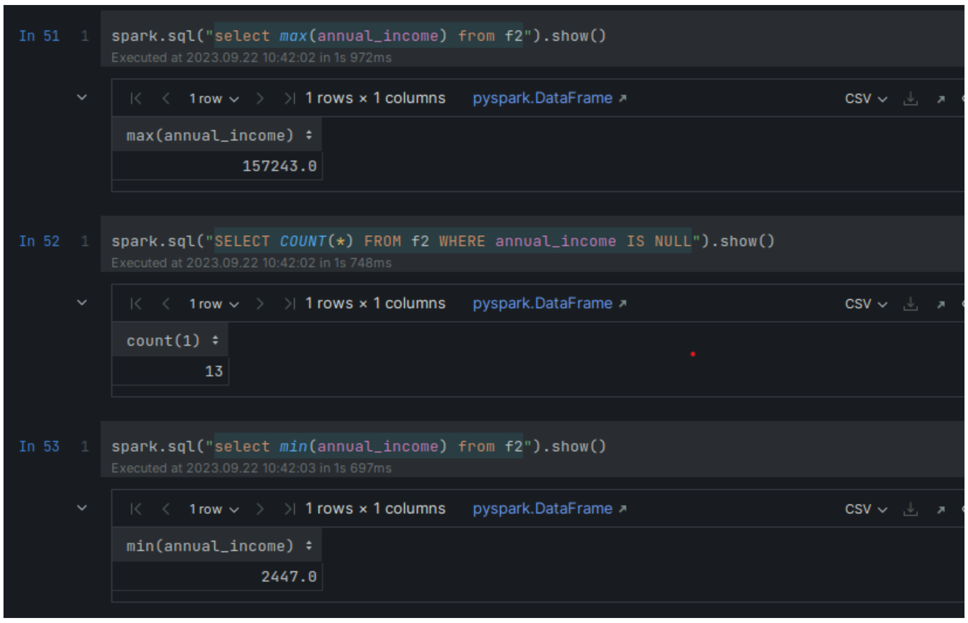
```
spark.sql("select online_purchases from f2 where annual_income>'50000'").show()
Executed at 2023.09.22 10:51:23 in 567ms
```

```
1-11   20 rows × 1 columns   pyspark.DataFrame   CSV

online_purchases
               8
               8
               5
               6
               7
               3
               6
               7
```

```
spark.sql("select annual_income from f2 where store_purchases>='5'").show()
Executed at 2023.09.22 10:42:01 in 1s 511ms
```

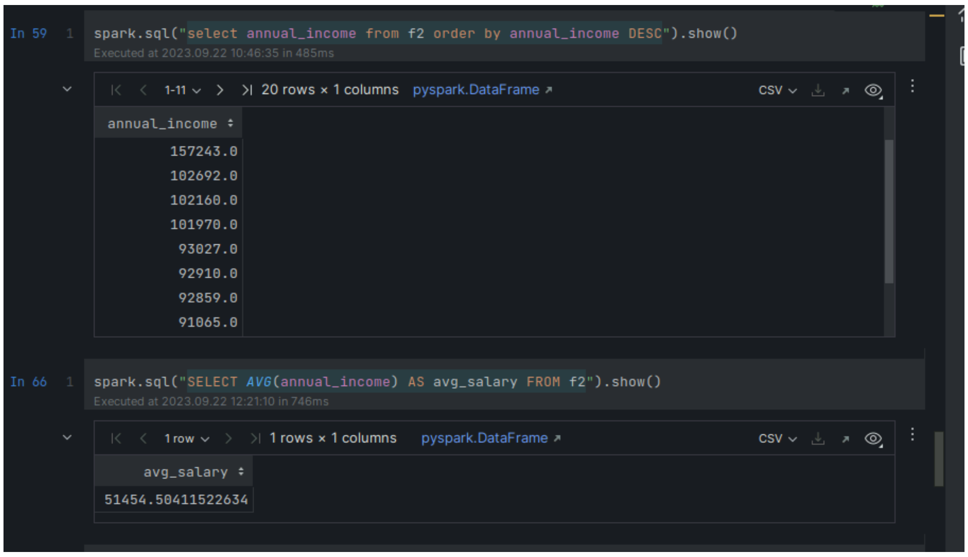
```
1-11   20 rows × 1 columns   pyspark.DataFrame   CSV

annual_income
      58293.0
      55635.0
      63033.0
      59354.0
      37760.0
```

9.



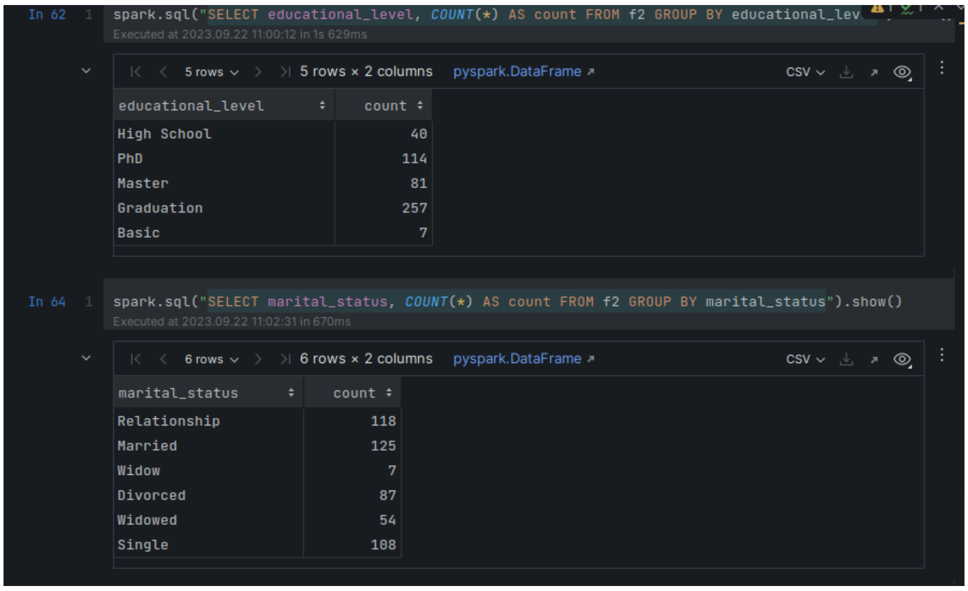
```
In 51   spark.sql("select max(annual_income) from f2").show()
        Executed at 2023.09.22 10:42:02 in 1s 972ms

        1 row   1 rows × 1 columns   pyspark.DataFrame   CSV
        max(annual_income)
                  157243.0

In 52   spark.sql("SELECT COUNT(*) FROM f2 WHERE annual_income IS NULL").show()
        Executed at 2023.09.22 10:42:02 in 1s 748ms

        1 row   1 rows × 1 columns   pyspark.DataFrame   CSV
        count(1)
              13

In 53   spark.sql("select min(annual_income) from f2").show()
        Executed at 2023.09.22 10:42:03 in 1s 697ms

        1 row   1 rows × 1 columns   pyspark.DataFrame   CSV
        min(annual_income)
                    2447.0
```

10.



```
In 59   spark.sql("select annual_income from f2 order by annual_income DESC").show()
        Executed at 2023.09.22 10:46:35 in 485ms

        1-11   20 rows × 1 columns   pyspark.DataFrame   CSV
        annual_income
             157243.0
             102692.0
             102160.0
             101970.0
              93027.0
              92910.0
              92859.0
              91065.0

In 66   spark.sql("SELECT AVG(annual_income) AS avg_salary FROM f2").show()
        Executed at 2023.09.22 12:21:10 in 746ms

        1 row   1 rows × 1 columns   pyspark.DataFrame   CSV
              avg_salary
        51454.50411522634
```

11.



```
In 62  spark.sql("SELECT educational_level, COUNT(*) AS count FROM f2 GROUP BY educational_lev
Executed at 2023.09.22 11:00:12 in 1s 629ms
```

5 rows × 2 columns pyspark.DataFrame

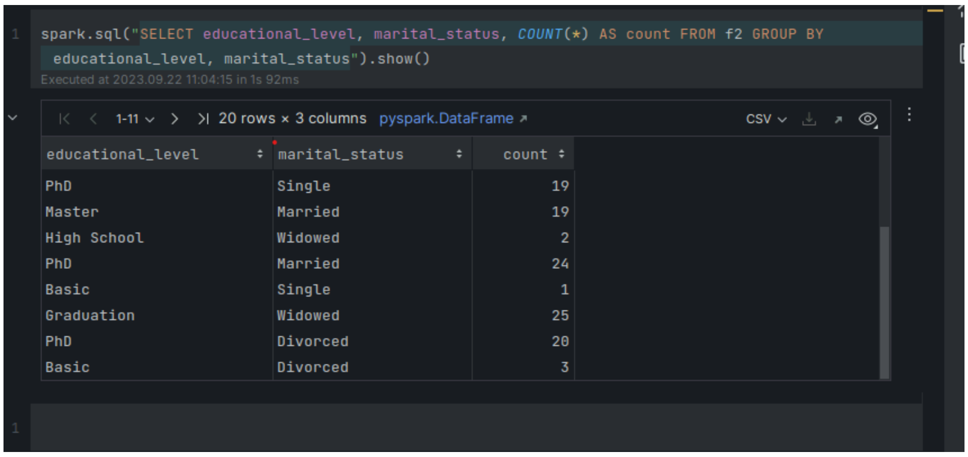
```
educational_level    count
High School             40
PhD                    114
Master                  81
Graduation             257
Basic                    7
```

```
In 64  spark.sql("SELECT marital_status, COUNT(*) AS count FROM f2 GROUP BY marital_status").show()
Executed at 2023.09.22 11:02:31 in 670ms
```

6 rows × 2 columns pyspark.DataFrame

```
marital_status   count
Relationship       118
Married            125
Widow                7
Divorced            87
Widowed             54
Single             108
```

12.



```
spark.sql("SELECT educational_level, marital_status, COUNT(*) AS count FROM f2 GROUP BY
 educational_level, marital_status").show()
Executed at 2023.09.22 11:04:15 in 1s 92ms
```

1-11 20 rows × 3 columns pyspark.DataFrame

```
educational_level   marital_status   count
PhD                 Single              19
Master              Married             19
High School         Widowed              2
PhD                 Married             24
Basic               Single               1
Graduation          Widowed             25
PhD                 Divorced            20
Basic               Divorced             3
```