```python
from pyspark.sql.functions import *

df =
spark.read.format("delta").load("/FileStore/lending/silver/delta")

# Q1 - Loan Approval Rate
total = df.count()
approved = df.filter(col("loan_status") == "Fully Paid").count()
print(f"Q1 - Approval Rate: {approved/total:.2%}")
```

```
Q1 - Approval Rate: 47.63%
```

```python
# Q2 - Average loan amount by grade
df.groupBy("grade").agg(avg("loan_amnt").alias("AvgLoan")).orderBy("gr
ade").show()
```

```
+-----+------------------+
|grade|           AvgLoan|
+-----+------------------+
|    A|14603.343209545825|
|    B| 14173.33819852703|
|    C|15038.083317821778|
|    D| 15711.98300680591|
|    E|17453.078391907933|
|    F| 19124.64653110048|
|    G|20383.988740959896|
+-----+------------------+
```

```python
# Q3 - Charged off vs Fully paid
df.groupBy("loan_status").count().filter(col("loan_status").isin("Char
ged Off", "Fully Paid")).show()
```

```
+-----------+-------+
|loan_status|  count|
+-----------+-------+
| Fully Paid|1076751|
|Charged Off| 268558|
+-----------+-------+
```

```python
# Q4 - Loan purpose distribution
df.groupBy("purpose").count().orderBy(desc("count")).show()
```

```
+--------------------+-------+
|             purpose|  count|
+--------------------+-------+
|  debt_consolidation|1277790|
|         credit_card| 516926|
|    home_improvement| 150440|
|               other| 139413|
```

```
|      major_purchase|  50429|
|             medical|  27481|
|      small_business|  24659|
|                 car|  24009|
|            vacation|  15525|
|              moving|  15402|
|               house|  14131|
|             wedding|   2351|
|    renewable_energy|   1445|
|         educational|    412|
| I have eliminate...|      1|
| my cell is on a ...|      1|
| MD""... approx. ...|      1|
| and have never h...|      1|
| which was a swin...|      1|
| usually doubling...|      1|
+--------------------+-------+
only showing top 20 rows


# Q5 - Unique states
print("Q5 - Unique States:",
df.select("addr_state").distinct().count())

Q5 - Unique States: 271

# Q6 - Average interest rate by sub-grade
df.groupBy("sub_grade").agg(avg("int_rate").alias("AvgInterest")).orde
rBy("sub_grade").show()

+---------+------------------+
|sub_grade|       AvgInterest|
+---------+------------------+
|       A1| 5.600265353152358|
|       A2| 6.552336764326663|
|       A3| 7.094534597727951|
|       A4|  7.56023906377143|
|       A5| 8.195006086398775|
|       B1|  9.07855442353105|
|       B2| 9.974970186619458|
|       B3| 10.70501452316554|
|       B4|11.372777749955029|
|       B5|12.013544779307304|
|       C1|12.783782992806444|
|       C2|13.537631867967175|
|       C3|14.104219423653143|
|       C4| 14.87805884435167|
|       C5|15.768242893616625|
|       D1|  16.6581692689528|
|       D2|17.599961179165266|
```

```
|       D3| 18.38845569971858|
|       D4| 19.07287050056509|
|       D5|20.063590779420135|
+---------+------------------+
only showing top 20 rows


# Q7 - Correlation between loan amount and interest rate
print("Q7 - Correlation:", df.stat.corr("loan_amnt", "int_rate"))

Q7 - Correlation: 0.09808178013492619


# Q8 - Average annual income
df.select(avg("annual_inc").alias("AvgAnnualIncome")).show()

+-----------------+
|  AvgAnnualIncome|
+-----------------+
|77992.44637764242|
+-----------------+



# Q9 - Default rate by employment length
df.withColumn("default", when(col("loan_status") == "Charged Off",
1).otherwise(0))\
  .groupBy("emp_length").agg(avg("default").alias("DefaultRate")).orde
rBy("emp_length").show()

+----------+-------------------+
|emp_length|        DefaultRate|
+----------+-------------------+
|      NULL|0.12674660364023005|
|         2|0.11838900999626858|
|         3|0.11886939635856666|
|         4|0.11639398265070824|
|         5|0.11807613566407536|
|         6|0.11829130451728573|
|         7| 0.1253573547656292|
|         8|0.13164479839850296|
|         9|0.12767806536935575|
|        10|0.11103802782066965|
+----------+-------------------+



# Q10 - Home ownership distribution
df.groupBy("home_ownership").count().orderBy(desc("count")).show()

+--------------+-------+
|home_ownership|  count|
```

```
+--------------+-------+
|      MORTGAGE|1111449|
|          RENT| 894929|
|           OWN| 253057|
|           ANY|    996|
|         OTHER|    182|
|          NONE|     54|
|       2 years|      1|
+--------------+-------+
```

```
# Q11 - Average debt-to-income ratio by grade
df.groupBy("grade").agg(avg("dti").alias("AvgDTI")).orderBy("grade").s
how()
```

```
+-----+------------------+
|grade|            AvgDTI|
+-----+------------------+
|    A|16.239339044863325|
|    B|17.967428521872705|
|    C|19.552627592025633|
|    D|20.930404344941337|
|    E|21.550460016086568|
|    F| 21.67774207451392|
|    G|22.434566935616655|
+-----+------------------+
```

```
# Q12 - Default rate by term (36 vs 60 months)
df.withColumn("default", when(col("loan_status") == "Charged Off",
1).otherwise(0))\
  .groupBy("term").agg(avg("default").alias("DefaultRate")).show()
```

```
+----------+-------------------+
|      term|        DefaultRate|
+----------+-------------------+
| 36 months| 0.1014142533579665|
| 60 months|0.16178174075223456|
+----------+-------------------+
```

```
# Q13 - Grade with highest loss given default
df.filter(col("loan_status") == "Charged Off")\
  .groupBy("grade").agg(avg("loan_amnt").alias("AvgLoss")).orderBy(des
c("AvgLoss")).show(1)
```

```
+-----+-----------------+
|grade|          AvgLoss|
+-----+-----------------+
|    G|20495.47149122807|
+-----+-----------------+
```

```
only showing top 1 row
```

```
# Q14 - Credit history length and approval
df.withColumn("credit_history_length", datediff(current_date(),
col("earliest_cr_line")) / 365.25)\
  .groupBy("loan_status").agg(avg("credit_history_length").alias("AvgC
reditHistoryLength")).show()
```

```
+--------------------+----------------------+
|         loan_status|AvgCreditHistoryLength|
+--------------------+----------------------+
|          Fully Paid|                  NULL|
|             Default|                  NULL|
|     In Grace Period|                  NULL|
|Does not meet the...|                  NULL|
|         Charged Off|    15.485284052019164|
|  Late (31-120 days)|                  NULL|
|             Current|                  NULL|
|Does not meet the...|                  NULL|
|   Late (16-30 days)|                  NULL|
|            Oct-2015|                  NULL|
+--------------------+----------------------+
```

```
# Q15 - Most common loan purpose in high-risk loans
df.filter(col("loan_status") == "Charged Off")\
  .groupBy("purpose").count().orderBy(desc("count")).show(1)
```

```
+------------------+------+
|           purpose| count|
+------------------+------+
|debt_consolidation|165005|
+------------------+------+
only showing top 1 row
```

```
# Q16 - Interest rate by purpose and term
df.groupBy("purpose",
"term").agg(avg("int_rate").alias("AvgInterest")).orderBy("purpose").s
how()
```

```
+--------------------+----------+-----------+
|             purpose|      term|AvgInterest|
+--------------------+----------+-----------+
|                NULL| 60 months|      10.99|
|      After gradu...| 60 months|      17.56|
|  (Citi Bank) whi...| 36 months|      14.96|
|  Hilal Khalil Ho...| 36 months|      11.54|
|  I have requeste...| 36 months|      13.85|
|  I realize that ...| 36 months|      13.79|
```

```
|  I'll work on th...| 36 months|       8.94|
|  and another one...| 36 months|      15.65|
|  but a bit too m...| 36 months|       9.45|
| 20 foot Yamaha s...| 36 months|        8.0|
| 5 years).  I req...| 36 months|       7.51|
|                768| 36 months|       7.88|
| BUT I LOVE THE C...| 36 months|      19.41|
| Bank of America ...| 36 months|      16.32|
| Butler PA. It is...| 36 months|       9.32|
| CA for 13 years....| 36 months|       9.38|
| CB Radio or Poli...| 36 months|      14.42|
|         Credit Card| 36 months|       9.45|
| I accidentally c...| 36 months|      13.57|
| I also always pa...| 36 months|      13.92|
+--------------------+----------+-----------+
only showing top 20 rows
```

```
# Q17 - Trend in number of loans over the years
df.withColumn("year", substring(col("issue_d"), -4, 4))\
  .groupBy("year").count().orderBy("year").show()
```

```
+----+------+
|year| count|
+----+------+
|2007|   603|
|2008|  2393|
|2009|  5281|
|2010| 12537|
|2011| 21721|
|2012| 53367|
|2013|134814|
|2014|235629|
|2015|421094|
|2016|434407|
|2017|443579|
|2018|495242|
|fied|     1|
+----+------+
```

```
# Q18 - Loans funded but not accepted (use proxy if available)
df.filter(col("loan_status").like("Does not meet%")).count()
```

```
2749
```

```
# Q19 - Average open credit lines
df.select(avg("open_acc").alias("AvgOpenCredit")).show()
```

```
+------------------+
|     AvgOpenCredit|
```

```
+------------------+
|11.624544599950363|
+------------------+


# Q20 - Top 5 employment titles among defaulters
df.filter(col("loan_status") == "Charged Off")\
  .groupBy("emp_title").count().orderBy(desc("count")).show(5)

+---------+-----+
|emp_title|count|
+---------+-----+
|     NULL|22461|
|  Manager| 4090|
|  Teacher| 3950|
|    Owner| 2696|
|   Driver| 2116|
+---------+-----+
only showing top 5 rows


# Q21 - Average revolving balance
df.select(avg("revol_bal").alias("AvgRevolvingBalance")).show()

+-------------------+
|AvgRevolvingBalance|
+-------------------+
|  16657.35474391925|
+-------------------+


# Q22 - Borrowers with no delinquencies
df.filter(col("delinq_2yrs") == 0).count()

1838880

# Q23 - Effect of verification status on loan defaults
df.withColumn("default", when(col("loan_status") == "Charged Off",
1).otherwise(0))\
  .groupBy("verification_status").agg(avg("default").alias("DefaultRat
e")).show()

+-------------------+-------------------+
|verification_status|        DefaultRate|
+-------------------+-------------------+
|           Verified|  0.158492831515602|
|    Source Verified|0.12325355720298342|
|       Not Verified|0.07993356659317997|
|            38000.0|                0.0|
+-------------------+-------------------+
```

```
# Q24 - Charge-off rate by loan amount buckets
df.withColumn("bucket", (floor(col("loan_amnt")/5000)*5000))\
  .withColumn("default", when(col("loan_status") == "Charged Off",
1).otherwise(0))\
  .groupBy("bucket").agg(avg("default").alias("ChargeOffRate")).orderB
y("bucket").show()
```

```
+------+-------------------+
|bucket|      ChargeOffRate|
+------+-------------------+
|     0|0.09602713810424686|
|  5000|0.10861329485018821|
| 10000| 0.1211712959624748|
| 15000|0.13309742080849735|
| 20000| 0.1309787315946492|
| 25000|0.12498709664099758|
| 30000|0.12193161748655568|
| 35000|0.13302418900797428|
| 40000| 0.0422260848717334|
+------+-------------------+
```

```
# Q25 - Average funded amount vs requested
df.select(avg("funded_amnt"), avg("loan_amnt")).show()
```

```
+------------------+------------------+
|  avg(funded_amnt)|    avg(loan_amnt)|
+------------------+------------------+
|15041.664056818605|15046.931227849467|
+------------------+------------------+
```

```
# Q26 - Proportion of jointly owned loans
df.groupBy("application_type").count().show()
```

```
+-----------------+------+
| application_type| count|
+-----------------+------+
|         Oct-2016|     1|
|              1.0|    57|
|            675.0|     2|
|         Jan-2012|     1|
|         May-2010|     1|
|         Jul-2010|     1|
|        Joint App|120710|
|         Jun-2011|     1|
|            710.0|     2|
|            660.0|     1|
|8541.663712072801|     1|
|         Feb-2011|     1|
|            645.0|     1|
```

```
|            509.0|     1|
|         May-2011|     1|
|             25.0|     1|
|         14153.43|     1|
|7393.367160475102|     1|
|             NULL|    55|
|                f|     1|
+-----------------+------+
only showing top 20 rows
```

```
# Q27 - Inquiries in last 6 months vs default
df.withColumn("default", when(col("loan_status") == "Charged Off",
1).otherwise(0))\
  .groupBy("inq_last_6mths").agg(avg("default").alias("DefaultRate")).
orderBy("inq_last_6mths").show()
```

```
+-------------------+-------------------+
|      inq_last_6mths|        DefaultRate|
+-------------------+-------------------+
|               NULL|0.03225806451612903|
|  I am currently ...|                0.0|
| $300.00 each.  P...|                0.0|
| AND QUIETLY.<br/...|                1.0|
| I have CreditSec...|                0.0|
| I need the reduc...|                0.0|
| I opted to partn...|                0.0|
| I will be able t...|                0.0|
| Marketing Materials|                0.0|
| and plenty of bu...|                0.0|
| and therefore re...|                0.0|
|               auto|                0.0|
| cancelling these...|                0.0|
| community and gr...|                0.0|
| household expens...|                0.0|
| however the rate...|                0.0|
| nearly all inves...|                1.0|
| or in full upon ...|                1.0|
|               rent|                0.0|
| so we know exact...|                0.0|
+-------------------+-------------------+
only showing top 20 rows
```

```
# Q28 - Common purposes for rejected loans
df_rejected = df.filter(col("loan_status") == "Rejected")
display(
    df_rejected.groupBy("purpose")
               .count()
```

```
                .orderBy("count", ascending=False)
)

# Q29 - Average installment by loan term
df.groupBy("term").agg(avg("installment").alias("AvgInstallment")).sho
w()
```

```
+----------+-----------------+
|      term|   AvgInstallment|
+----------+-----------------+
| 36 months|422.1533858216925|
| 60 months|504.3033599369498|
+----------+-----------------+
```

```
# Q30 - Top state in terms of loan origination

from pyspark.sql.functions import count

display(
    df.groupBy("addr_state")
      .agg(count("*").alias("LoanCount"))
      .orderBy("LoanCount", ascending=False)
      .limit(1)
)

# Q31 - Average income-to-loan ratio
from pyspark.sql.functions import col, avg

display(
    df.withColumn("income_to_loan_ratio", col("annual_inc") /
col("loan_amnt"))
      .agg(avg("income_to_loan_ratio").alias("AvgIncomeToLoanRatio"))
)

# Q32 - Default distribution among top 10 zip codes

from pyspark.sql.functions import count

top_zip_codes = (
    df.groupBy("zip_code")
      .agg(count("*").alias("LoanCount"))
      .orderBy("LoanCount", ascending=False)
      .limit(10)
      .select("zip_code")
)

default_distribution = (
    df.join(top_zip_codes, on="zip_code", how="inner")
      .groupBy("zip_code", "loan_status")
      .agg(count("*").alias("Count"))
```

```
      .orderBy("zip_code", "loan_status")
)

display(default_distribution)

# Q33 - Loans with 0 credit lines in last 2 years (proxy using `pub_rec`)
loans_with_zero_credit_lines = df.filter(col("pub_rec") == 0)
display(loans_with_zero_credit_lines)

# Q34 - Employment length distribution in defaulted loans
employment_length_distribution = (
    df.filter(col("loan_status") == "Default")
      .groupBy("emp_length")
      .agg(count("*").alias("Count"))
      .orderBy("emp_length")
)

display(employment_length_distribution)
```