"PAY_PATTERNS: INSIGHTS INTO LOAN REPAYMENT"

OWNER - DIVYANSHU SRIVASTAVA

In the SQL notebook, we successfully performed data quality checks, cleaning, and integrity validation, ensuring that our dataset is accurate, consistent, and analysis-ready.

Now, in this Jupyter notebook, we take the next step – bridging SQL with Python. By connecting directly to our MySQL database, we'll run targeted SQL queries to extract meaningful insights. These insights will then be transformed into interactive and visually compelling plots using Matplotlib and Seaborn, making complex patterns easy to understand and communicate.

This notebook serves as the analytical storytelling layer of our project—where raw numbers evolve into clear narratives, empowering data-driven decisions.

In [1]:
```
# Importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import mysql.connector
import getpass
```

In [2]:
```
# Function to connect to the datbase
def create_connection():
    """
    Create a secure connection to the MySQL database.
    Uses getpass for password input (avoids hardcoding).
    """
    try:
        # Ask for password securely
        password = getpass.getpass("Enter MySQL password: ")

        # Database configuration
        db_config = {
            'user': 'root',
            'password': password,
            'host': 'localhost',
            'database': 'loan_analytics'
        }

        # Establish connection
        conn = mysql.connector.connect(**db_config)
        if conn.is_connected():
            print("Connected to MySQL Database!")
            return conn
    except mysql.connector.Error as e:
        print("Error while connecting:", e)
        return None
```

In [3]:
```
# Connecting to the database
conn = create_connection()
```

```
Connected to MySQL Database!
```

In [4]:
```
# Function for running the SQL queries and storing the resut as a dataframe
def run_query(conn, query):
    """
    Run a SQL query using mysql-connector and return DataFrame.
    """
    try:
        cursor = conn.cursor(dictionary = True)  # results as dict
        cursor.execute(query)
        rows = cursor.fetchall()
        df = pd.DataFrame(rows) if rows else pd.DataFrame()
        cursor.close()
        return df
    except Exception as e:
        print("Error running query:", e)
        return None
```

BASIC DESCREPTIVE STATS

1. BORROWERS TABLE-

The Borrowers table reveals strong diversity across demographics and credit profiles. Annual incomes range from very low to extremely high, with an average of ~₹80k, highlighting a wide borrower spectrum. Most borrowers hold around 25 total credit accounts, indicating established credit histories. Employment length shows concentration in the 3–5 year range, reflecting mid-career individuals forming the bulk of applicants.

In terms of stability, nearly half of borrowers own homes with a mortgage, while 41% rent and a smaller share own their homes outright. Importantly, about two-thirds of borrowers have verified incomes, showing that lenders rely significantly on verification while still extending credit to a notable non-verified group. Together, these patterns suggest lenders are serving a mix of mid-income, mid-career borrowers, balancing risk through income verification and collateral (homeownership), while still leaving exposure in unverified and rental-heavy segments.

Check minimum and maximum, average annual income of borrowers.

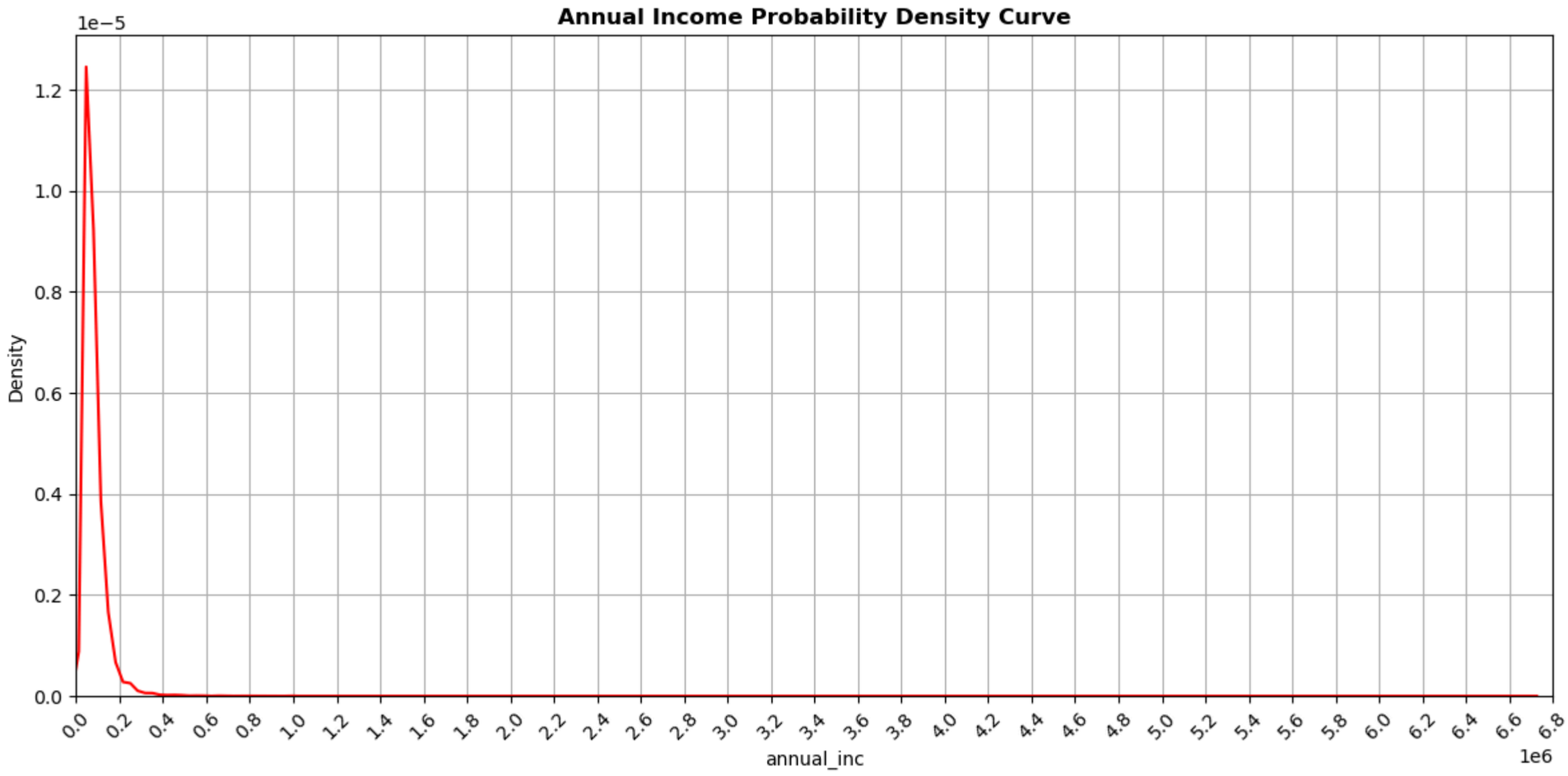
Borrower incomes in the portfolio range widely—from as low as $2,400 to as high as over $6.7 million, with an average around $80K across nearly 50,000 borrowers. The KDE plot shows a heavy concentration of incomes at the lower to middle range, while only a few outliers push the maximum upwards. This wide spread highlights that while most borrowers come from modest income brackets, a small fraction of very high earners significantly skews the distribution

In [5]:
```
query = """
        SELECT MIN(annual_inc) AS min_income,
               MAX(annual_inc) AS max_income,
               AVG(annual_inc) AS avg_income,
               COUNT(*) AS income_entries
        FROM borrowers;
        """

df = run_query(conn,query)
print(df)
```

```
   min_income  max_income    avg_income  income_entries
0      2400.0   6702150.0  80378.688246           49950
```

In [6]:
```
query = """
        SELECT annual_inc,borrower_id
        FROM borrowers;
        """

df = run_query(conn,query)
```

```
plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'annual_inc',color = 'r')
plt.xticks(ticks = range(0,6800001,200000))
plt.xlim(0,6800001)
plt.xticks(rotation = 45)
plt.title('Annual Income Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```



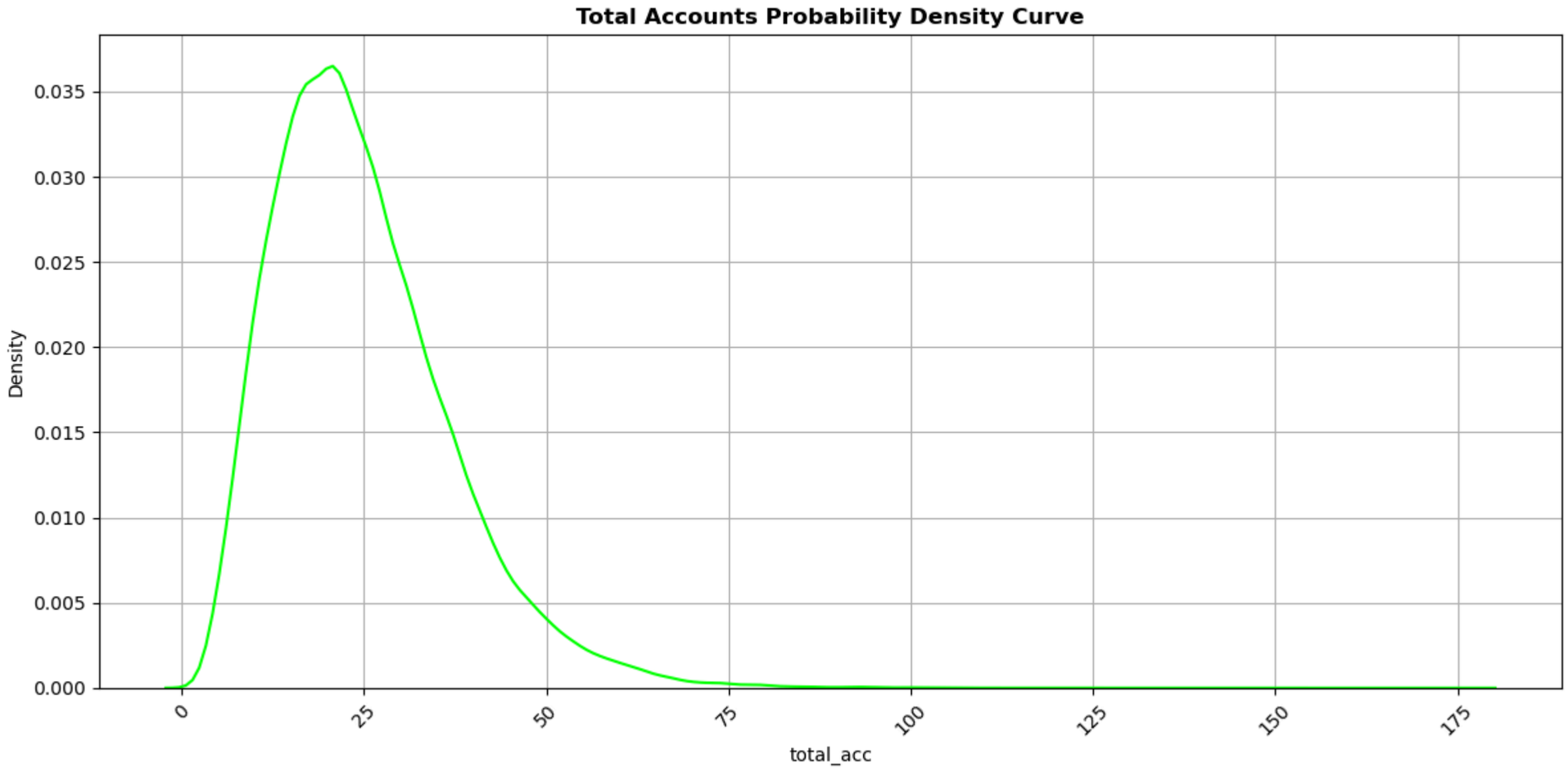
Check distribution of total accounts among borrowers.

Borrowers hold anywhere between 2 and 176 credit accounts, with an average of about 25 accounts. The density plot shows that most borrowers cluster in the lower-to-mid account range, while a few outliers with extremely high numbers of accounts stretch the distribution’s tail. This suggests that while the typical borrower manages a moderate number of accounts, a small subset may be significantly more leveraged, potentially carrying higher complexity in repayment behavior.

In [7]:
```
query = """
       SELECT MIN(total_acc) AS min_acc,
       MAX(total_acc) AS max_acc,
       AVG(total_acc) AS avg_acc,
       COUNT(*) AS total_acc_entries
       FROM borrowers;
"""

df = run_query(conn,query)
print(df)
```

```
   min_acc  max_acc  avg_acc  total_acc_entries
0        2      176  24.7088              49950
```

In [8]:
```
query = """
        SELECT total_acc,borrower_id
        FROM borrowers;
        """

df = run_query(conn,query)

plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'total_acc',color = 'lime')
plt.xticks(rotation = 45)
plt.title('Total Accounts Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```

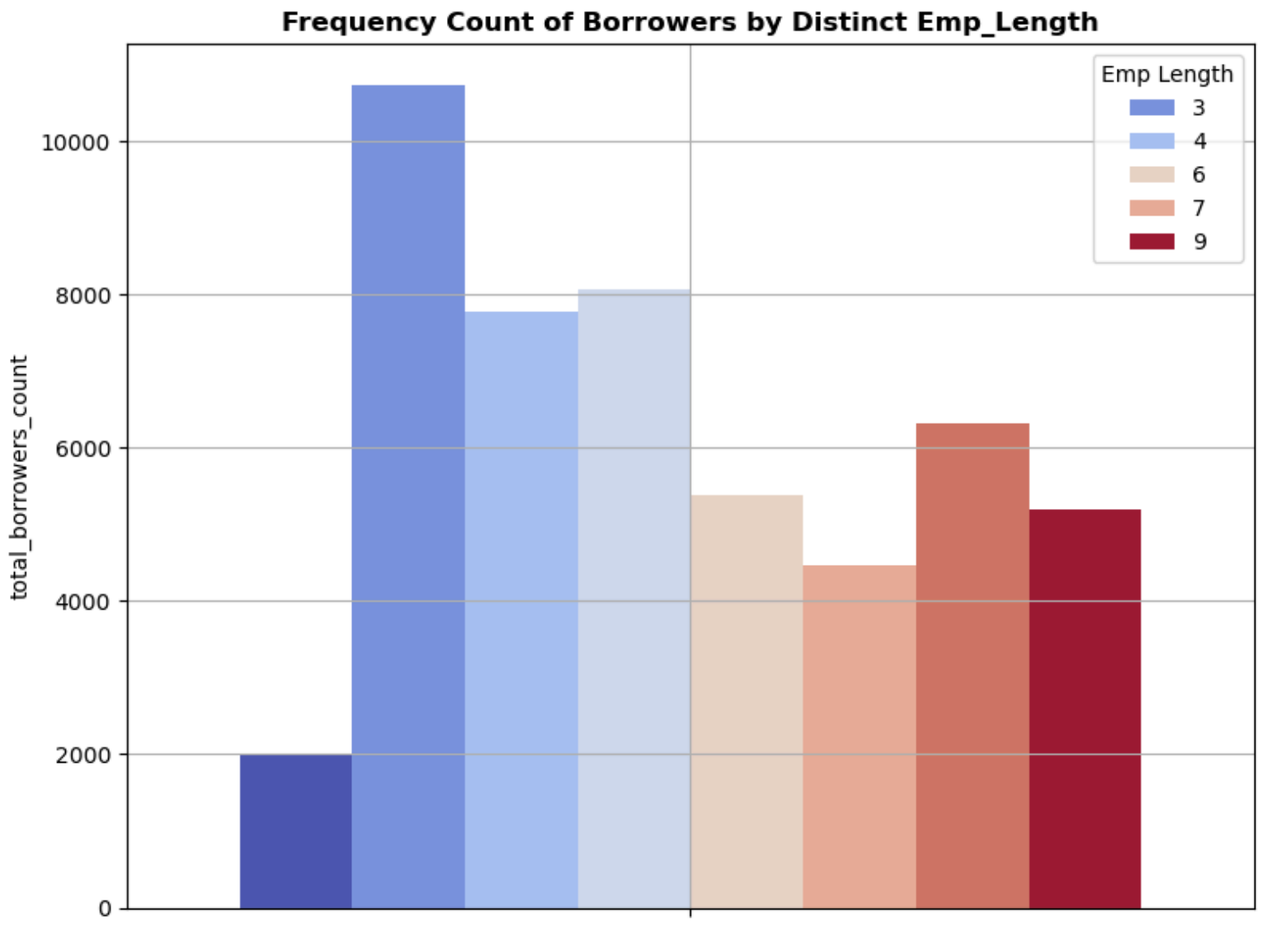


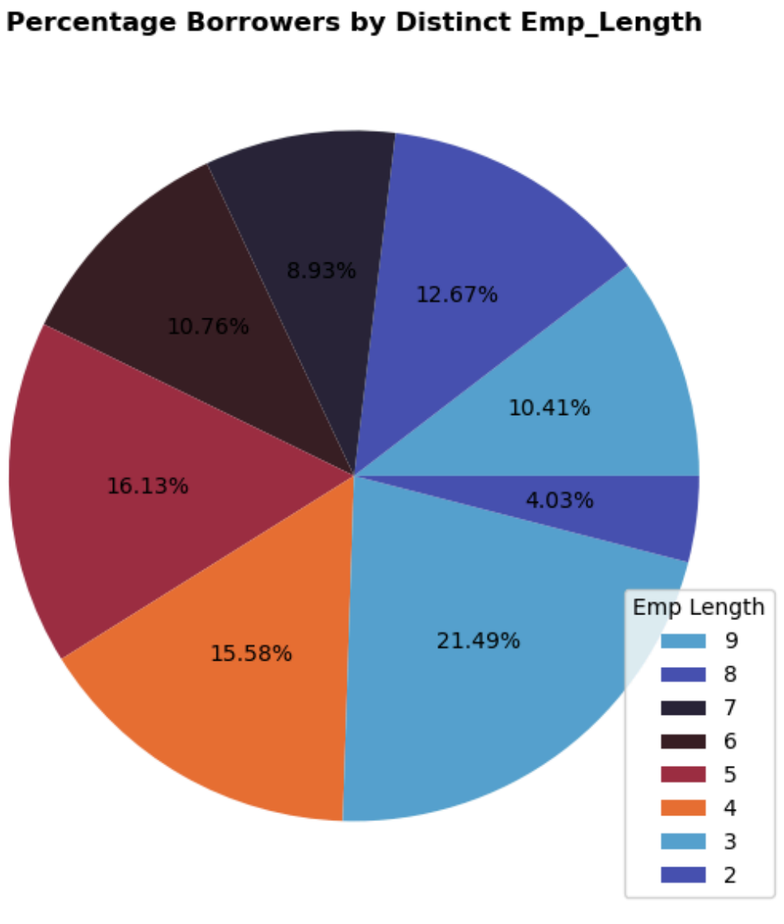
Check number of borrowers by distinct employment length.

The borrower base is fairly diverse in employment history, but the majority cluster around 3–5 years of experience, together making up over 50% of the pool. Shorter tenures like 2 years or less account for only a small slice, while longer stability of 8–9 years is also present but less common. This distribution highlights that lenders are primarily dealing with borrowers in the mid-stage of their careers, suggesting moderate income stability but not always the long-term track record associated with very low credit risk.

In [9]:
```
query = """
      SELECT  emp_length,COUNT(*) AS 'total_borrowers_count',
      ROUND(COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers),2) AS 'percent_borrowers'
      FROM borrowers
      GROUP BY emp_length
      ORDER BY emp_length DESC;
"""

df = run_query(conn,query)
print(df,end = '\n\n')

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'emp_length', y = 'total_borrowers_count',palette = 'coolwarm',ax = ax[0])
ax[0].legend(title = "Emp Length")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Distinct Emp_Length',fontweight = 'bold')

ax[1].pie(df['percent_borrowers'],autopct = '%1.2f%%',colors = sns.color_palette('icefire'))
ax[1].legend(labels = df['emp_length'],title="Emp Length")
ax[1].set_title('Percentage Borrowers by Distinct Emp_Length',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
   emp_length  total_borrowers_count  percent_borrowers
0           9                   5199              10.41
1           8                   6328              12.67
2           7                   4463               8.93
3           6                   5374              10.76
4           5                   8059              16.13
5           4                   7782              15.58
6           3                  10732              21.49
7           2                   2013               4.03
```



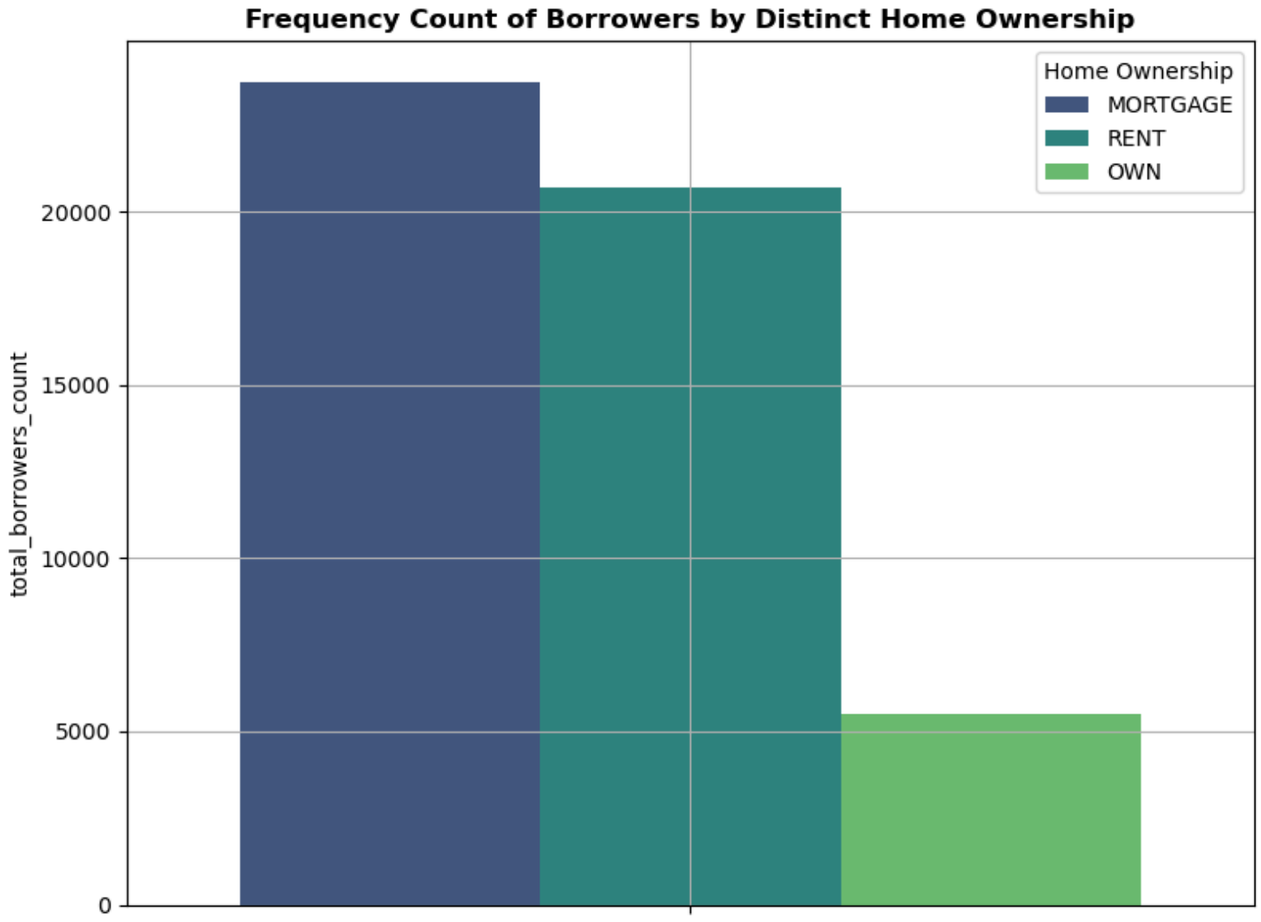


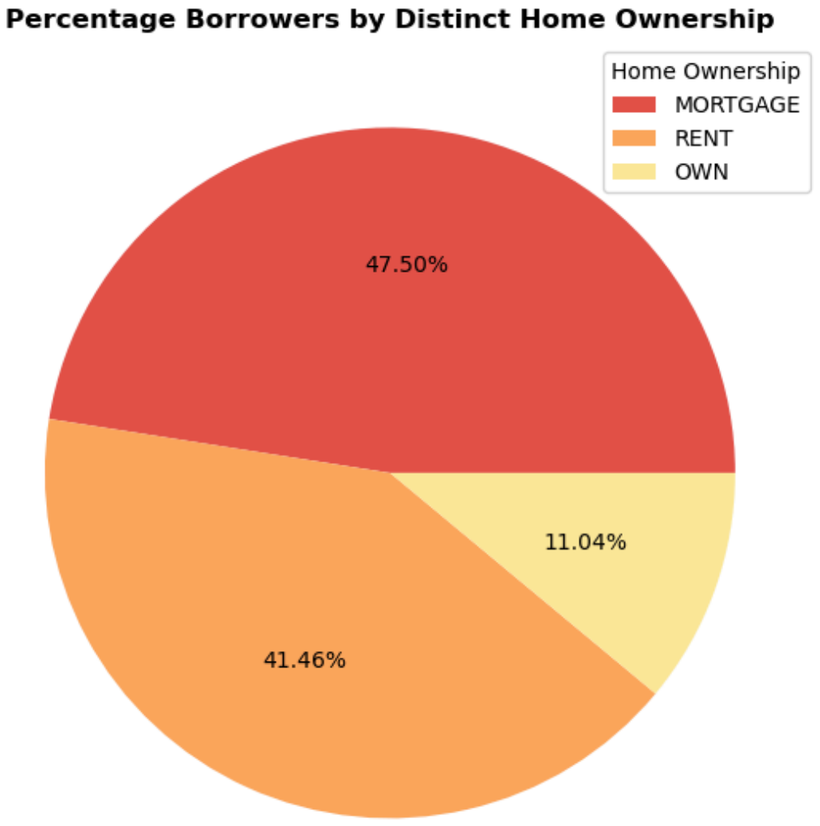
Check number of borrowers by distinct home ownership values

Nearly half of all borrowers are financing their homes through a mortgage (47.5%), while another 41% live in rentals, showing that a large portion of borrowers do not fully own their homes yet. Only about 11% are outright owners, a much smaller segment. This split suggests that most borrowers are still in the wealth-building stage, with significant housing costs either as rent or mortgage, which could impact repayment capacity but also signals steady financial commitments.

In [10]:
```
query = """SELECT home_ownership,  COUNT(*) AS 'total_borrowers_count',
            COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_borrowers'
            FROM borrowers
            GROUP BY home_ownership
            ORDER BY total_borrowers_count DESC;"""

df = run_query(conn,query)
print(df,end = '\n\n')

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'home_ownership', y = 'total_borrowers_count',palette = 'viridis',ax = ax[0])
ax[0].legend(title="Home Ownership")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Distinct Home Ownership',fontweight = 'bold')

ax[1].pie(df['percent_borrowers'],autopct = '%1.2f%%',colors = sns.color_palette('Spectral'))
ax[1].legend(labels = df['home_ownership'],title = "Home Ownership")
ax[1].set_title('Percentage Borrowers by Distinct Home Ownership',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
  home_ownership  total_borrowers_count  percent_borrowers
0       MORTGAGE                  23726            47.4995
1           RENT                  20711            41.4635
2            OWN                   5513            11.0370
```



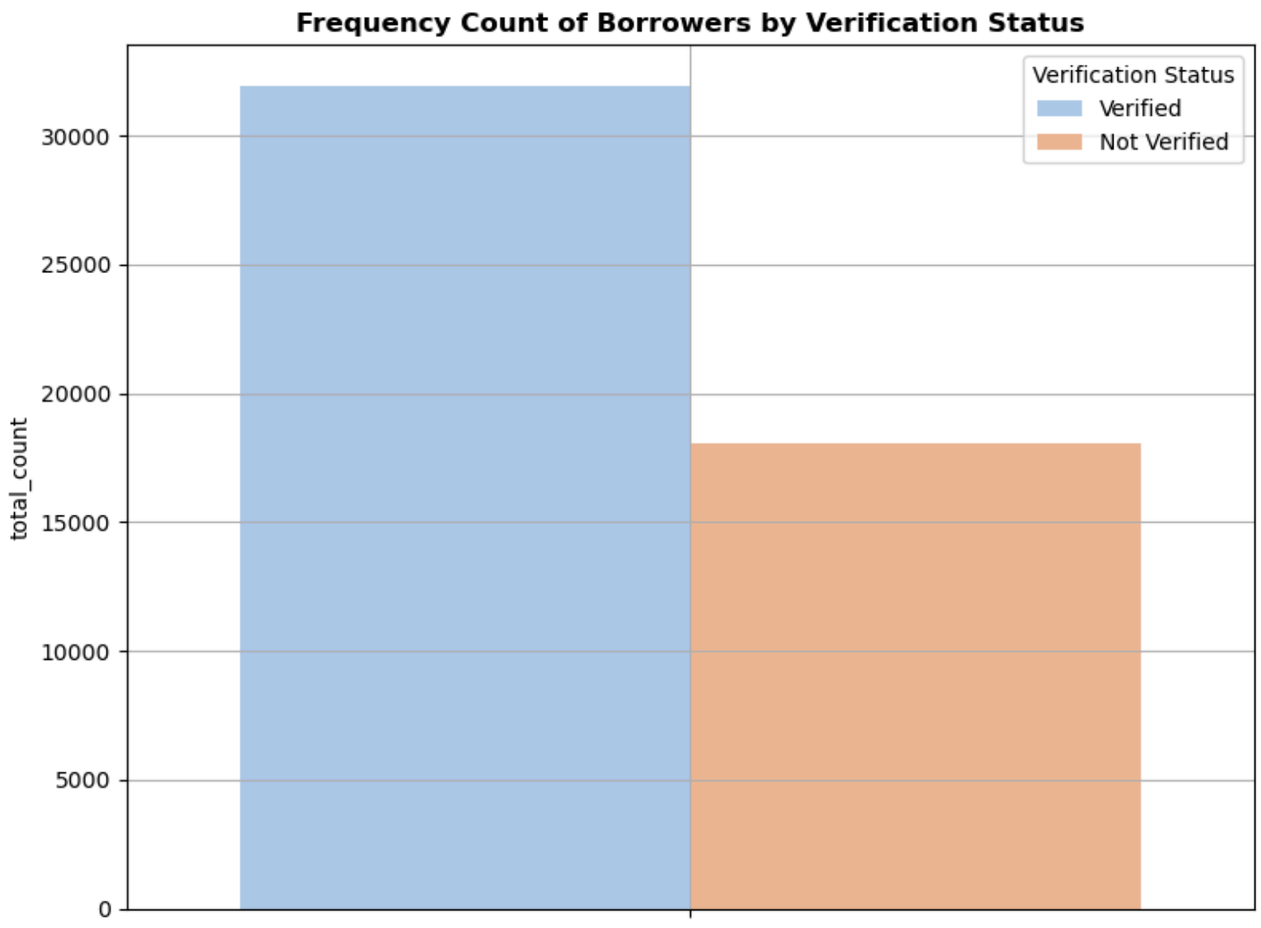


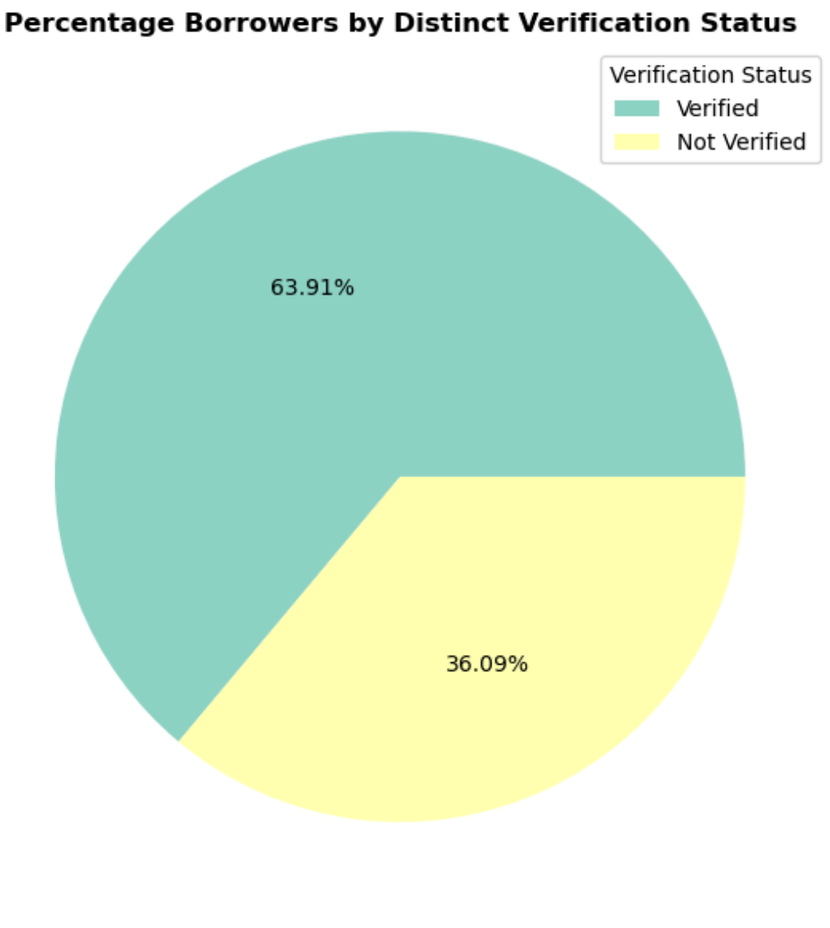
Check total borrowers distinct verification status values

Around 64% of borrowers have their income verified, while the remaining 36% are not verified. This indicates that lenders rely heavily on verified income as a safeguard, but a significant chunk of loans is still issued without strict verification. In real-world terms, this split highlights a trade-off: verified borrowers provide more confidence in repayment ability, whereas non-verified borrowers may pose higher risk but are still included to expand lending opportunities.

In [11]:
```
query = """SELECT DISTINCT verification_status, COUNT(*) AS 'total_count',
           COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_borrowers'
           FROM borrowers
           GROUP BY verification_status
           ORDER BY total_count DESC;"""

df = run_query(conn,query)
print(df,end='\n\n')

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'verification_status', y = 'total_count',palette = 'pastel',ax = ax[0])
ax[0].legend(title="Verification Status")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Verification Status',fontweight = 'bold')

ax[1].pie(df['percent_borrowers'],autopct = '%1.2f%%',colors = sns.color_palette('Set3'))
ax[1].legend(labels = df['verification_status'],title = "Verification Status")
ax[1].set_title('Percentage Borrowers by Distinct Verification Status',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
  verification_status  total_count percent_borrowers
0            Verified        31924           63.9119
1        Not Verified        18026           36.0881
```





2. LOANS TABLE-

The loans dataset shows that most borrowers take loans between $1,000 and $40,000, with an average loan of ~$15,800, and prefer 36–60 month terms. Interest rates vary widely from 5.3% to 29%, with an average of 12.5%, reflecting both low- and high-risk lending. Debt-to-income ratios are generally moderate (avg ~19.4%), but some borrowers have extremely high leverage. Most borrowers have low delinquency history, though a few exhibit repeated late payments, and revolving credit utilization is broad, averaging 52%, indicating varying credit behavior.

Borrower segmentation by grade shows that grades B and C dominate, while higher-risk grades F and G are much smaller in share. Regarding loan purpose, debt consolidation and credit card refinancing are most common, comprising the bulk of the portfolio. Loan status analysis indicates that while a majority of borrowers are current or fully paid, around 25% have charged-off loans, highlighting risk exposure. Overall, the loans table reflects a diverse portfolio with concentration in mid-grade, debt consolidation, and credit card loans, along with key areas of credit risk to monitor.

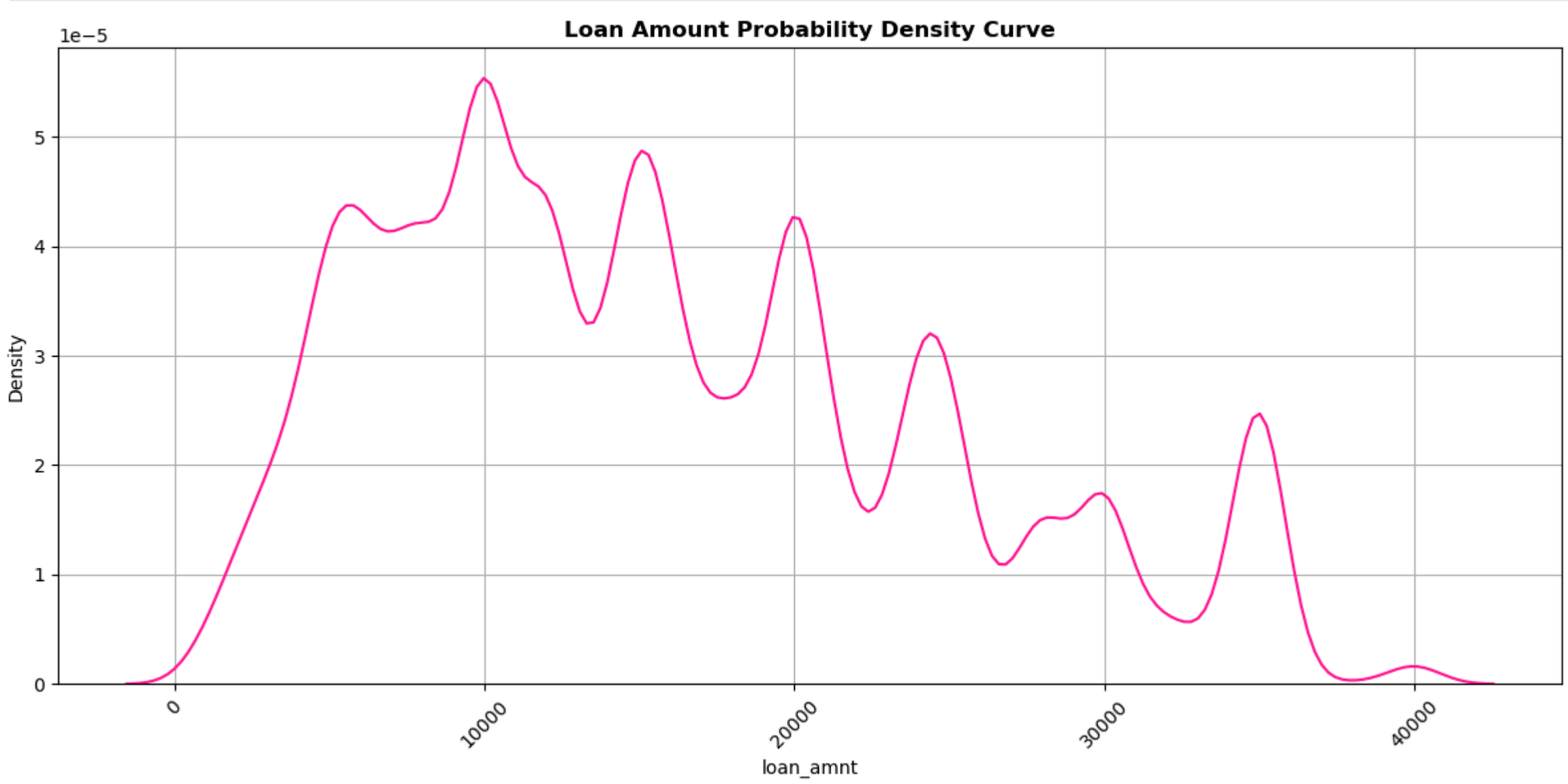
Check range of loan_amnt

The loan amounts issued to borrowers range from as little as $1,000 to $40,000, with an average of around $15,800. The KDE plot shows a clear clustering toward the mid-to-lower range, suggesting that most borrowers seek moderate loan sizes rather than the maximum limits. This reflects typical borrowing behavior, where loans are often used for debt consolidation or manageable personal expenses, while very high-value loans remain relatively rare.

In [12]:
```
query = """SELECT MIN(loan_amnt) AS min_loan,
           MAX(loan_amnt) AS max_loan,
           AVG(loan_amnt) AS avg_loan
           FROM loans;"""

df = run_query(conn,query)
print(df)
```

```
   min_loan  max_loan    avg_loan
0      1000     40000  15823.4753
```

In [13]:
```
query = """
        SELECT loan_amnt,loan_id
        FROM loans;
        """

df = run_query(conn,query)

plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'loan_amnt',color = 'deeppink')
plt.xticks(rotation = 45)
plt.title('Loan Amount Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Check range of term

Borrowers take loans with terms ranging from 36 to 60 months, with the average term settling at about 43 months. This indicates a stronger borrower preference for medium-duration repayment periods, balancing affordability of installments with overall interest burden.

In [14]:
```
query = """SELECT MIN(term) AS min_term,
           MAX(term) AS max_term,
           AVG(term) AS avg_term
           FROM loans;"""

df = run_query(conn,query)
print(df)
```

```
   min_term  max_term avg_term
0        36        60  42.9497
```

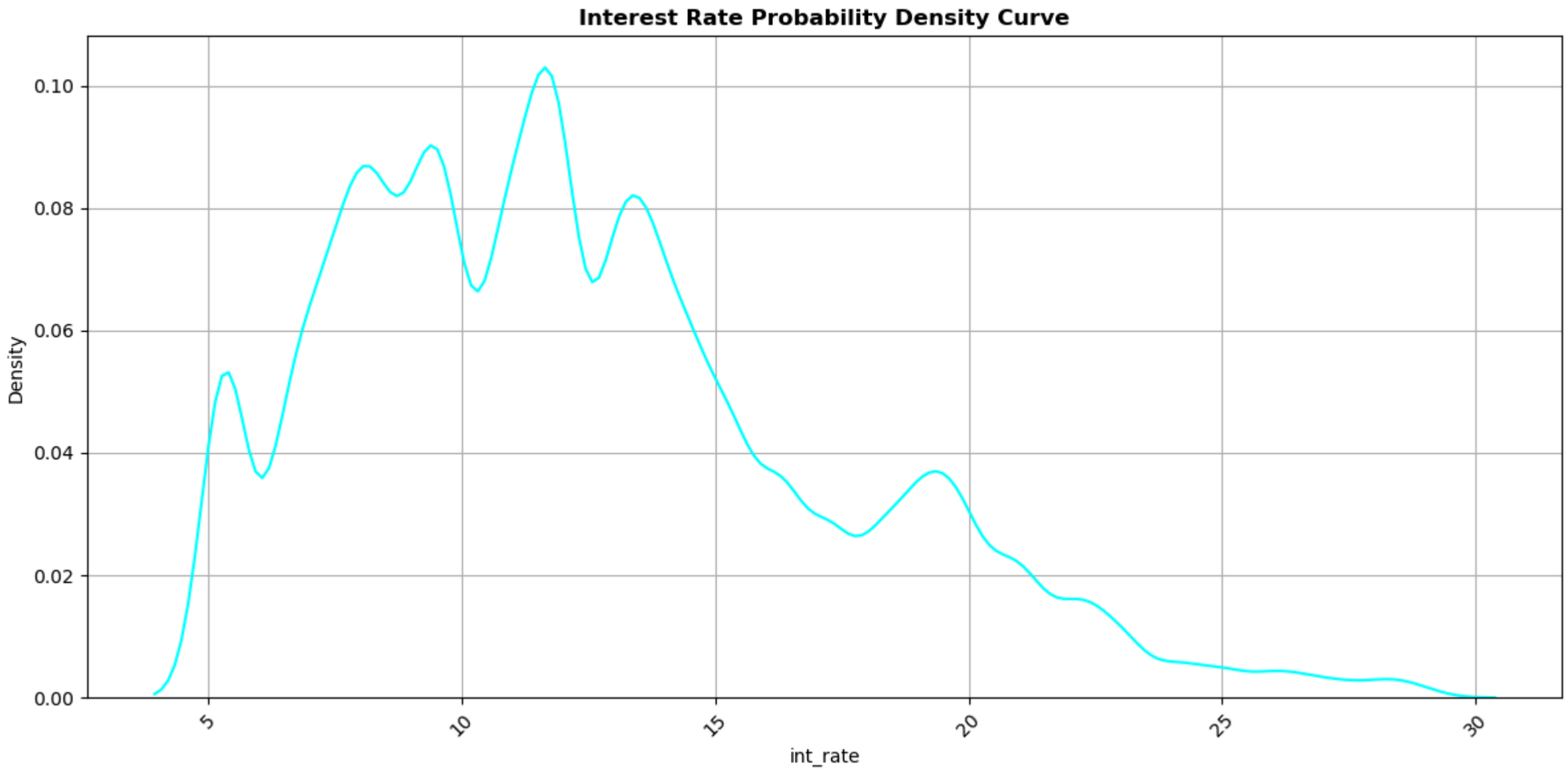
Check range of int_rate

The interest rates on loans range from 5.3% to nearly 29%, with an average around 12.5%. The KDE curve shows most loans clustering in the lower-to-mid interest range, highlighting that while affordable credit is accessible to many, a noticeable share of borrowers—likely riskier profiles—face substantially higher rates.

In [15]:
```
query = """SELECT MIN(int_rate) AS min_rate,
             MAX(int_rate) AS max_rate,
             AVG(int_rate) AS avg_rate
             FROM loans;"""

df = run_query(conn,query)
print(df)
```

```
   min_rate max_rate   avg_rate
0      5.32    28.99  12.467679
```

In [16]:
```
query = """
        SELECT int_rate,loan_id
        FROM loans;
        """

df = run_query(conn,query)

plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'int_rate',color = 'cyan')
plt.xticks(rotation = 45)
plt.title('Interest Rate Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```



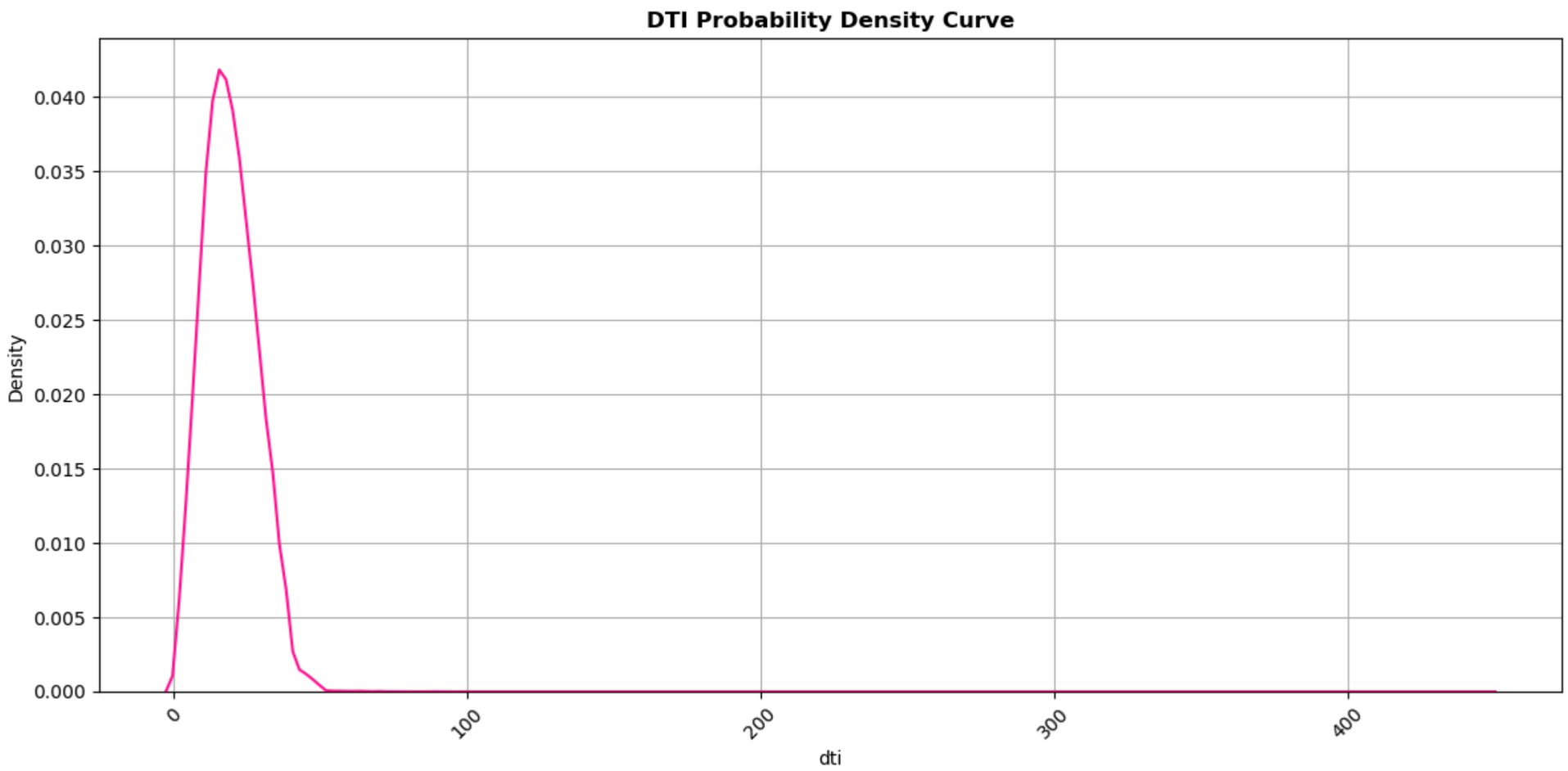
Check range of dti

The DTI ratio among borrowers spans from 0 up to a very high 447.6, though the average is around 19.4. The probability density curve reveals that most borrowers fall within a reasonable DTI band, while a small fraction carry extremely high debt burdens relative to income. These extreme outliers highlight potential financial stress and default risk, making DTI a critical factor in lending decisions.

In [17]:
```
query = """SELECT MIN(dti) AS min_dti,
           MAX(dti) AS max_dti,
           AVG(dti) AS avg_dti
           FROM loans;"""

df = run_query(conn,query)
print(df)
```

```
   min_dti  max_dti    avg_dti
0      0.0    447.6  19.395423
```

In [18]:
```
query = """
        SELECT dti,loan_id
        FROM loans;
        """

df = run_query(conn,query)

plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'dti',color = 'deeppink')
plt.xticks(rotation = 45)
plt.title('DTI Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Check range of delinq_2yrs

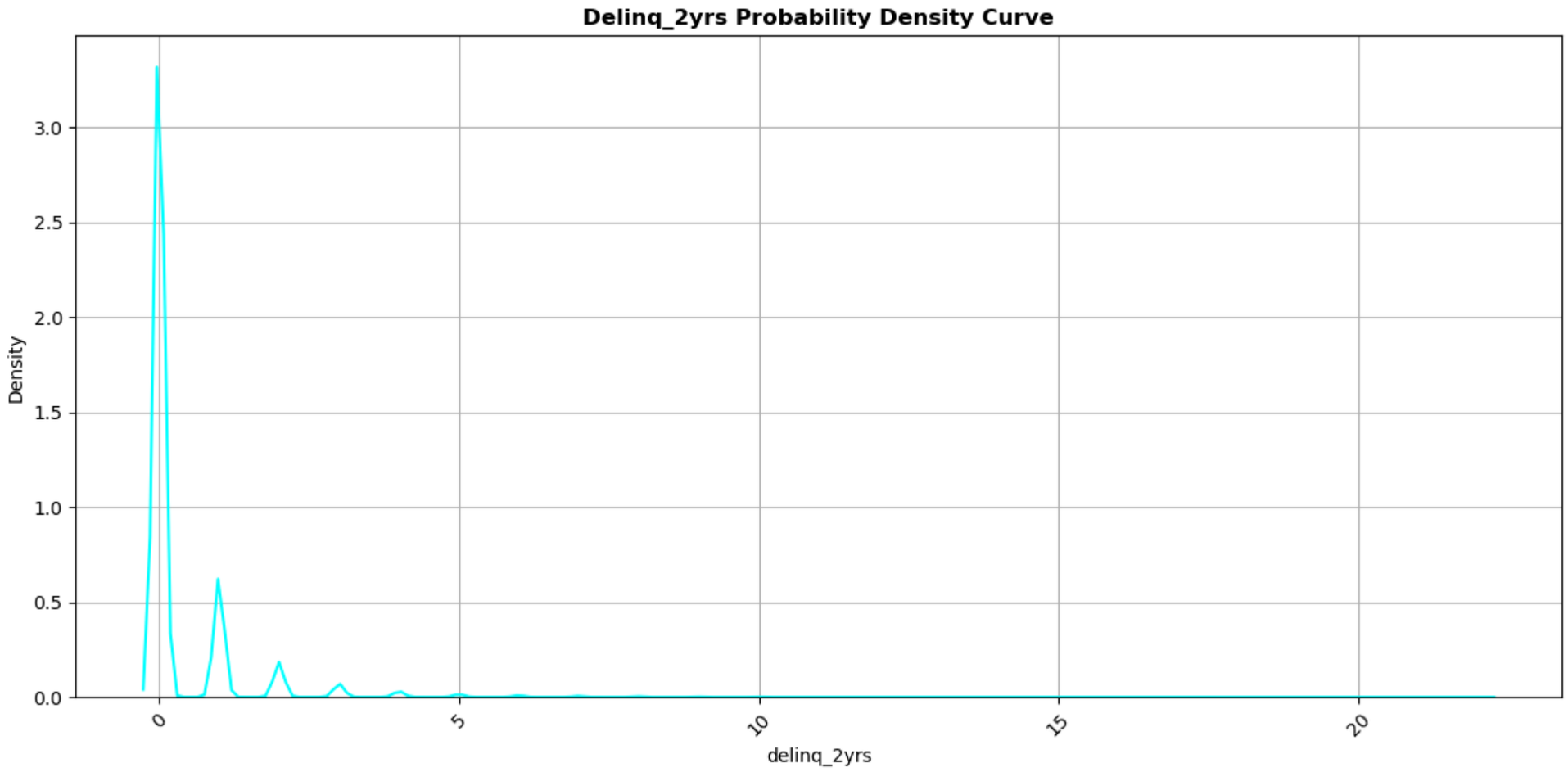
Most borrowers have zero delinquencies in the past 2 years, as reflected by the min and average values (0 and 0.348, respectively). However, a few borrowers have had as many as 22 delinquencies, creating a long-tailed distribution in the probability density curve. This shows that while the majority of borrowers are financially disciplined, a small fraction carry significant credit risk.

In [19]:
```
query = """SELECT MIN(delinq_2yrs) AS min_delinq_2yrs,
           MAX(delinq_2yrs) AS max_delinq_2yrs,
           AVG(delinq_2yrs) AS avg_delinq_2yrs
           FROM loans;"""

df = run_query(conn,query)
print(df)
```

```
   min_delinq_2yrs  max_delinq_2yrs avg_delinq_2yrs
0                0               22          0.3480
```

In [20]:
```
query = """
        SELECT delinq_2yrs,loan_id
```

```
        FROM loans;
        """

df = run_query(conn,query)

plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'delinq_2yrs',color = 'cyan')
plt.xticks(rotation = 45)
plt.title('Delinq_2yrs Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```



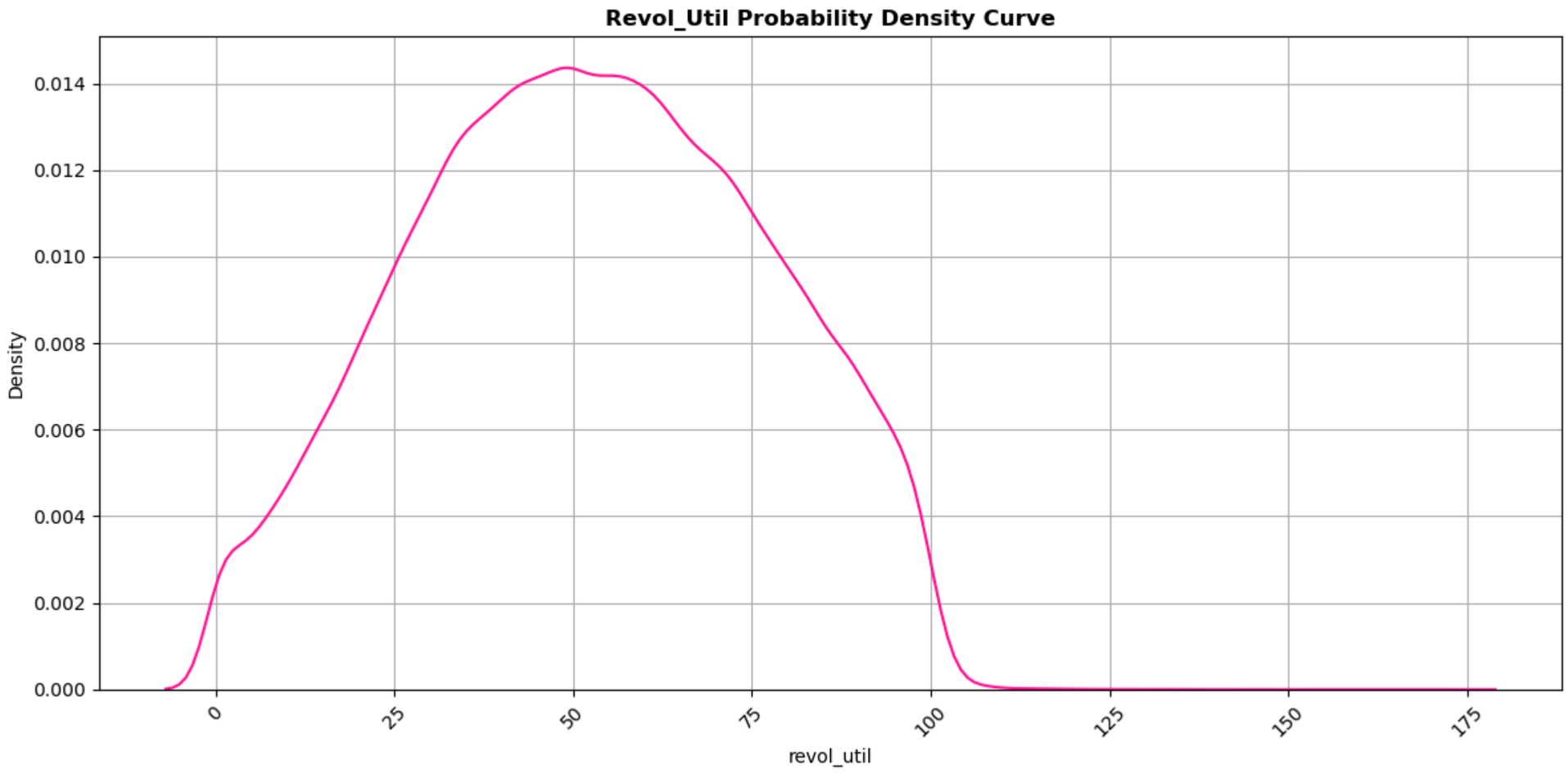
Check range of revol_util

The revolving credit utilization among borrowers shows a wide range, from 0% to an extreme 172%, with an average around 52%. The KDE curve highlights that most borrowers maintain moderate utilization, but some exceed their credit limits, indicating potential over-leveraging. This variable is key for assessing credit risk, as higher utilization generally signals higher likelihood of default.

In [21]:
```
query = """SELECT MIN(revol_util) AS min_revol_util,
           MAX(revol_util) AS max_revol_util,
           AVG(revol_util) AS avg_revol_util
           FROM loans;"""

df = run_query(conn,query)
print(df)
```

```
   min_revol_util  max_revol_util  avg_revol_util
0            0.00          172.00       52.178385
```

In [22]:
```
query = """
        SELECT revol_util,loan_id
        FROM loans;
        """

df = run_query(conn,query)

plt.figure(figsize = (12,6))
sns.kdeplot(data = df,x = 'revol_util',color = 'deeppink')
plt.xticks(rotation = 45)
plt.title('Revol_Util Probability Density Curve',fontweight = 'bold')
plt.grid(True)
plt.tight_layout()
plt.show()
```

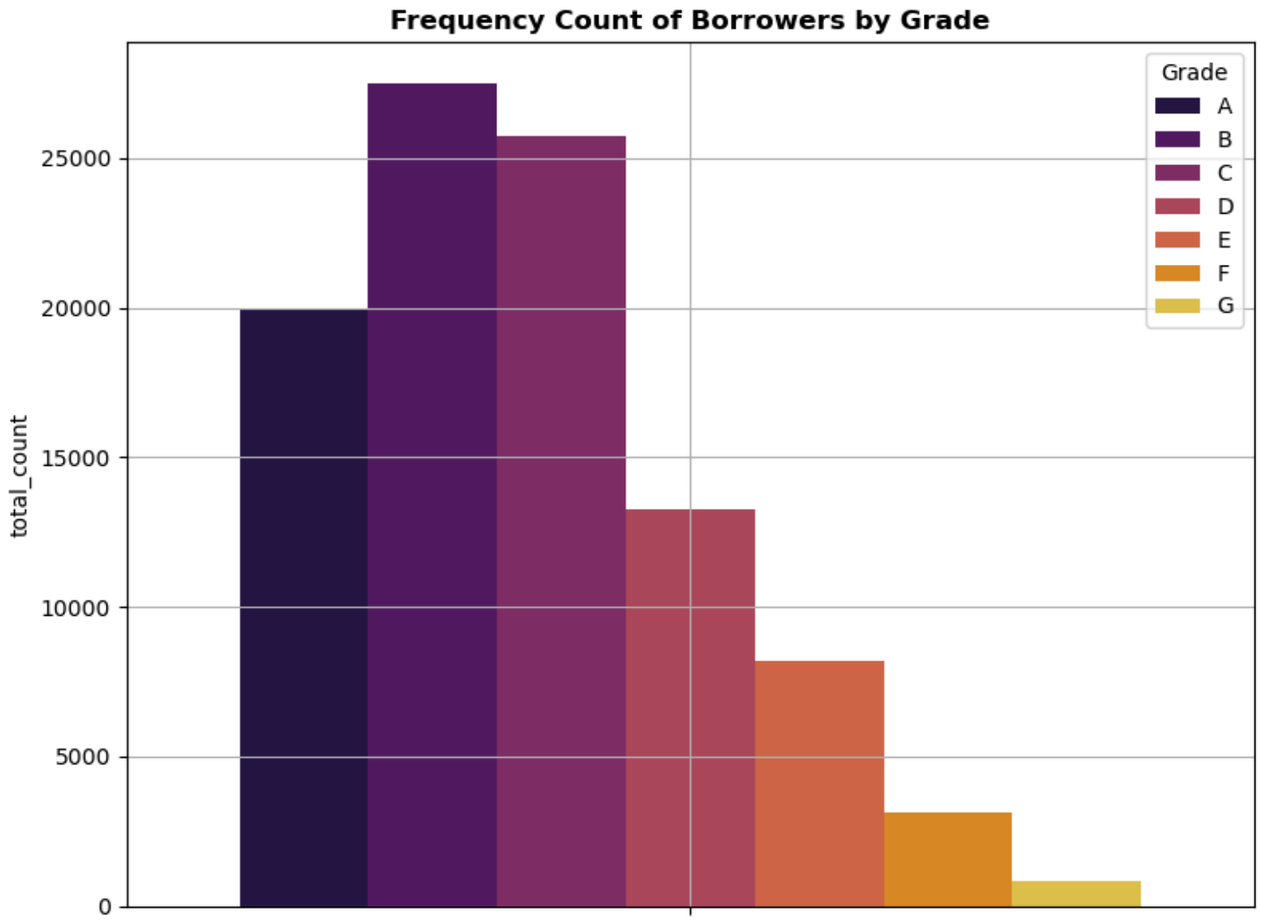


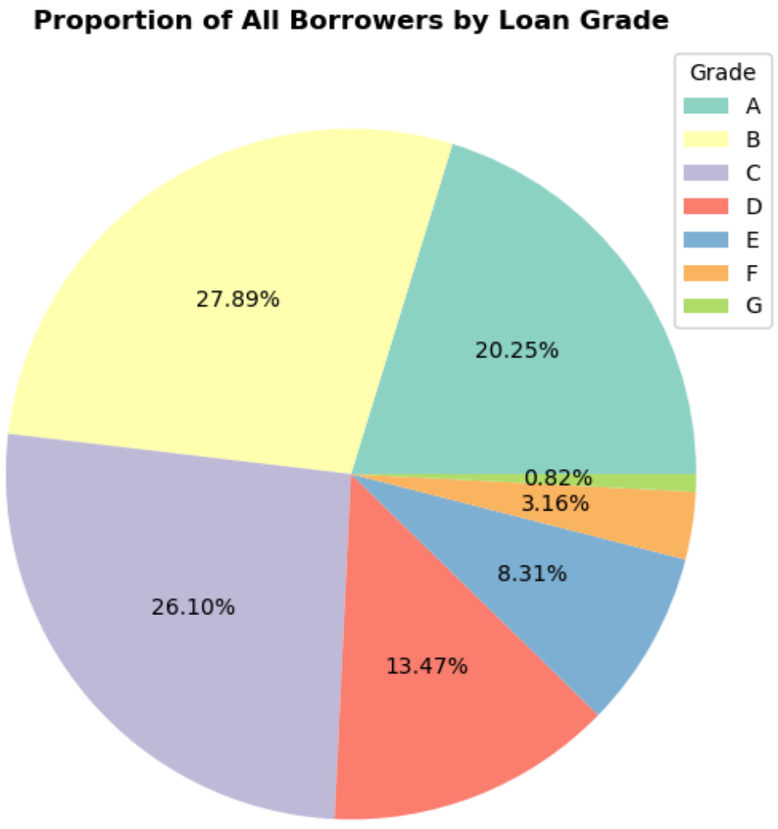
Check total borrowers by distinct grade values

The distribution of borrowers across loan grades is heavily skewed toward the mid-range grades, with B (55%) and C (52%) having the largest share of borrowers. Grades A and D–G have fewer borrowers, with the highest-risk grade G representing only about 1.6% of the total. The bar and pie plots together highlight that most borrowers fall into moderate-risk categories, while very high- or low-risk borrowers are comparatively rare, guiding lenders in portfolio targeting and risk management.

In [23]:
```
query = """SELECT grade, COUNT(DISTINCT(borrower_id)) AS 'total_count',
           COUNT(DISTINCT(borrower_id)) * 100/ (SELECT COUNT(borrower_id) FROM borrowers) AS 'per_grade_borrower_ratio'
           FROM loans
           GROUP BY grade
           ORDER BY grade;"""

df = run_query(conn,query)
print(df,end = '\n\n')

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'grade', y = 'total_count',palette = 'inferno',ax = ax[0])
ax[0].legend(title="Grade")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Grade',fontweight = 'bold')

ax[1].pie(df['per_grade_borrower_ratio'],autopct = '%1.2f%%',colors = sns.color_palette('Set3'))
ax[1].legend(labels = df['grade'],title = "Grade")
ax[1].set_title('Proportion of All Borrowers by Loan Grade',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
  grade  total_count per_grade_borrower_ratio
0     A        19964                 39.9680
1     B        27502                 55.0591
2     C        25736                 51.5235
3     D        13280                 26.5866
4     E         8193                 16.4024
5     F         3115                  6.2362
6     G          811                  1.6236
```



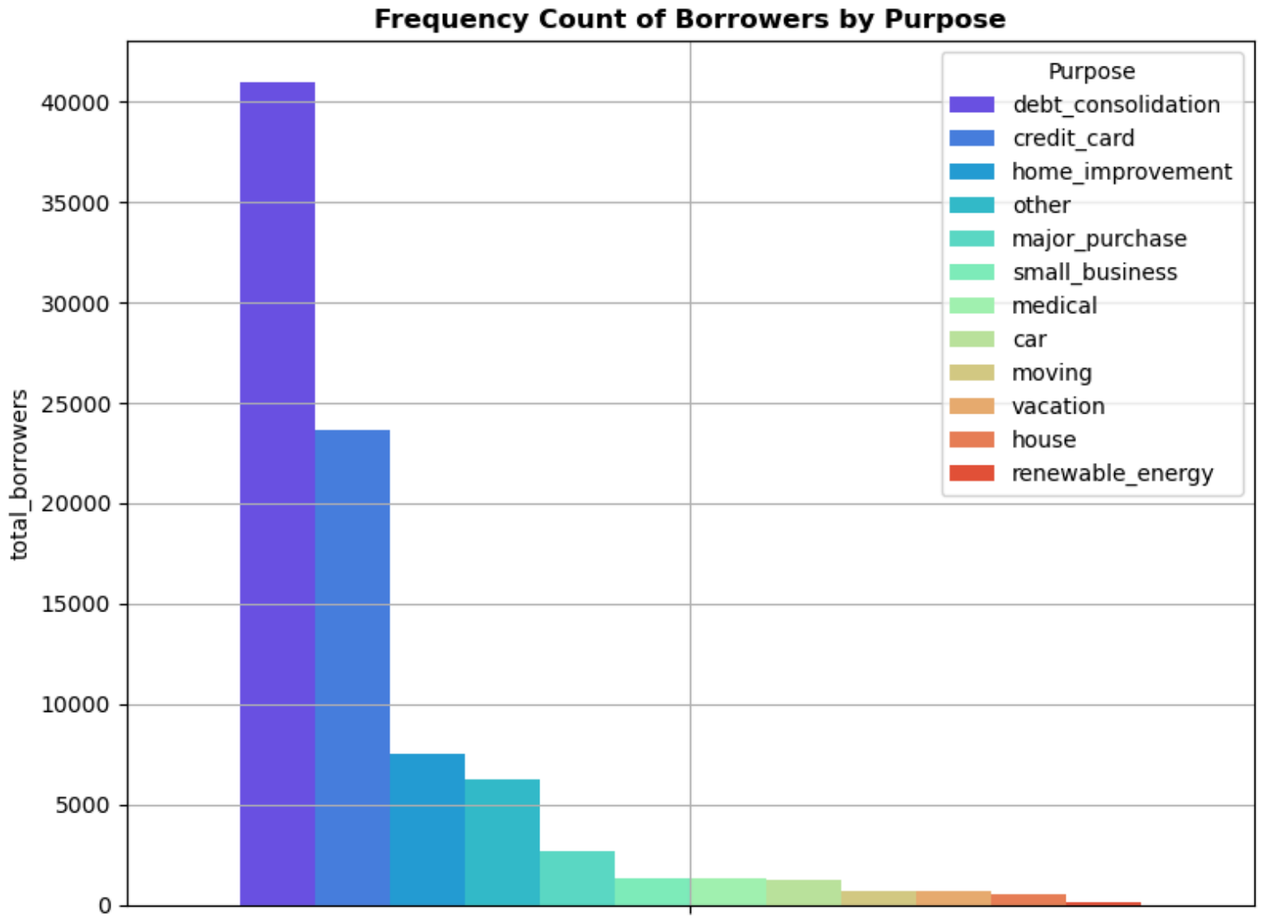


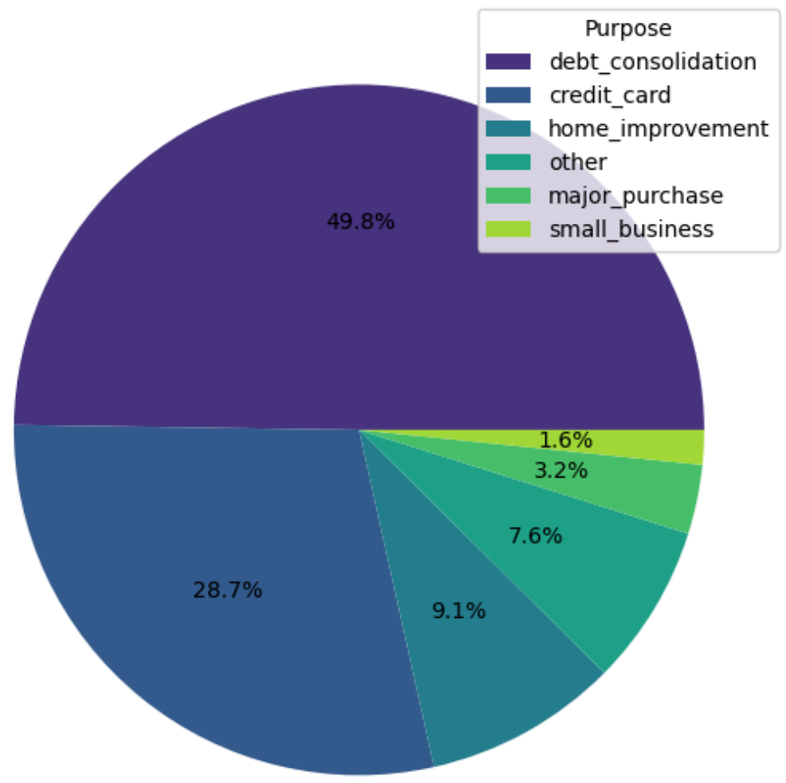
Check total borrowers by distinct purpose values

The majority of borrowers take loans for debt consolidation (82%) and credit card repayment (47%), making these two purposes dominant in the portfolio. Other purposes like home improvement, major purchases, or small business account for a much smaller share. The bar and pie plots emphasize that lenders are primarily exposed to borrowers managing existing debts, highlighting the importance of tailored risk assessment and repayment support strategies for these high-volume segments.

In [24]:
```
query = """SELECT purpose, COUNT(DISTINCT(borrower_id)) AS 'total_borrowers',
           COUNT(DISTINCT(borrower_id)) * 100/ (SELECT COUNT(borrower_id) FROM  borrowers) AS 'percent_borrowers_per_purpose'
           FROM loans
           GROUP BY purpose
           ORDER BY total_borrowers DESC;"""

df = run_query(conn,query)
print(df,end = '\n\n')

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'purpose', y = 'total_borrowers',palette = 'rainbow',ax = ax[0])
ax[0].legend(title="Purpose")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Purpose',fontweight = 'bold')

ax[1].pie(df['percent_borrowers_per_purpose'].head(6),autopct = '%1.1f%%',colors = sns.color_palette('viridis'))
ax[1].legend(labels = df['purpose'],title="Purpose")
ax[1].set_title('Proportion of All Borrowers by Distinct Purpose',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
              purpose  total_borrowers percent_borrowers_per_purpose
0   debt_consolidation            40970                       82.0220
1          credit_card            23644                       47.3353
2     home_improvement             7488                       14.9910
3                other             6222                       12.4565
4       major_purchase             2669                        5.3433
5       small_business             1331                        2.6647
6              medical             1301                        2.6046
7                  car             1211                        2.4244
8               moving              712                        1.4254
9             vacation              684                        1.3694
10               house              521                        1.0430
11    renewable_energy               82                        0.1642
```



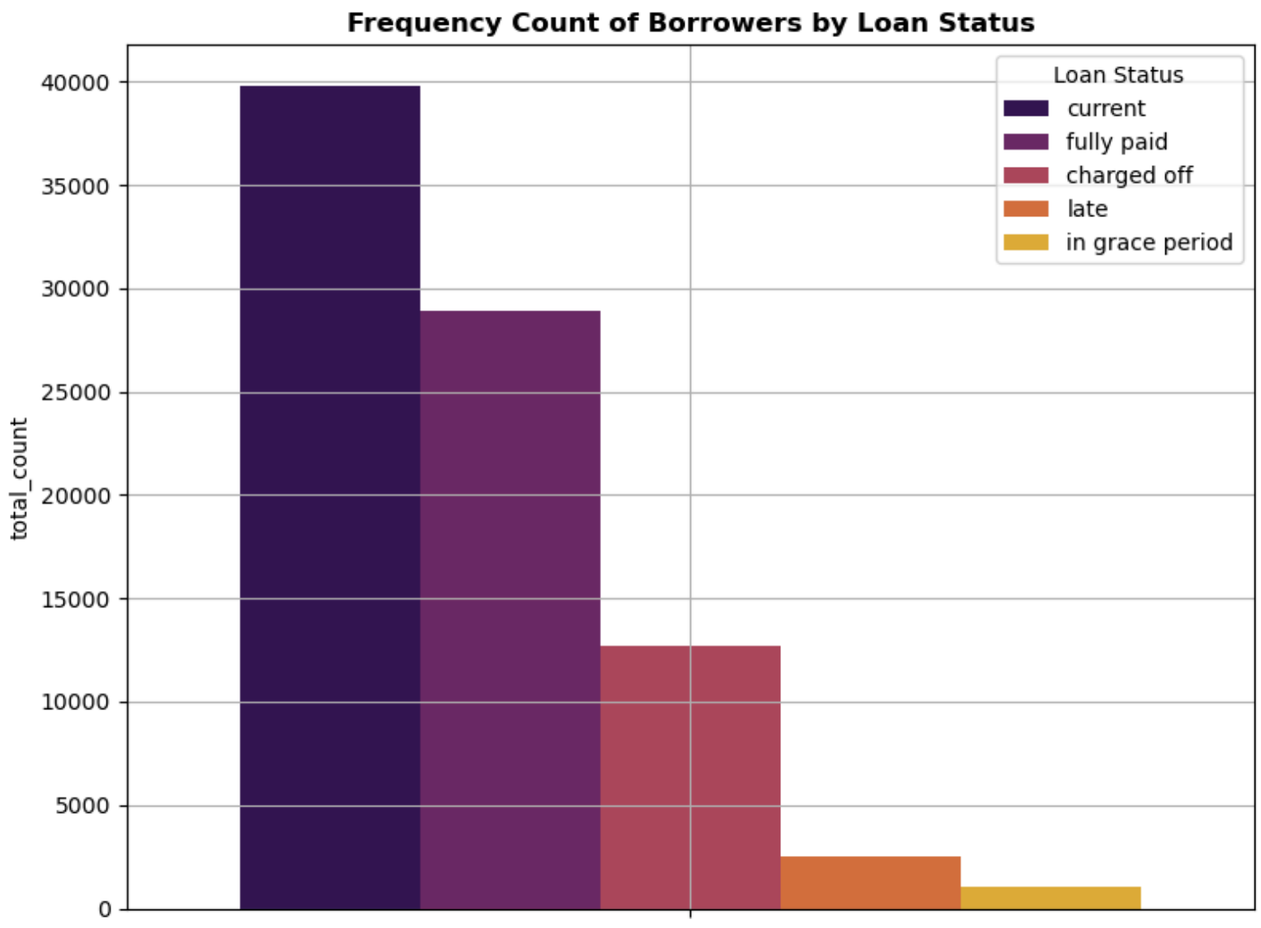


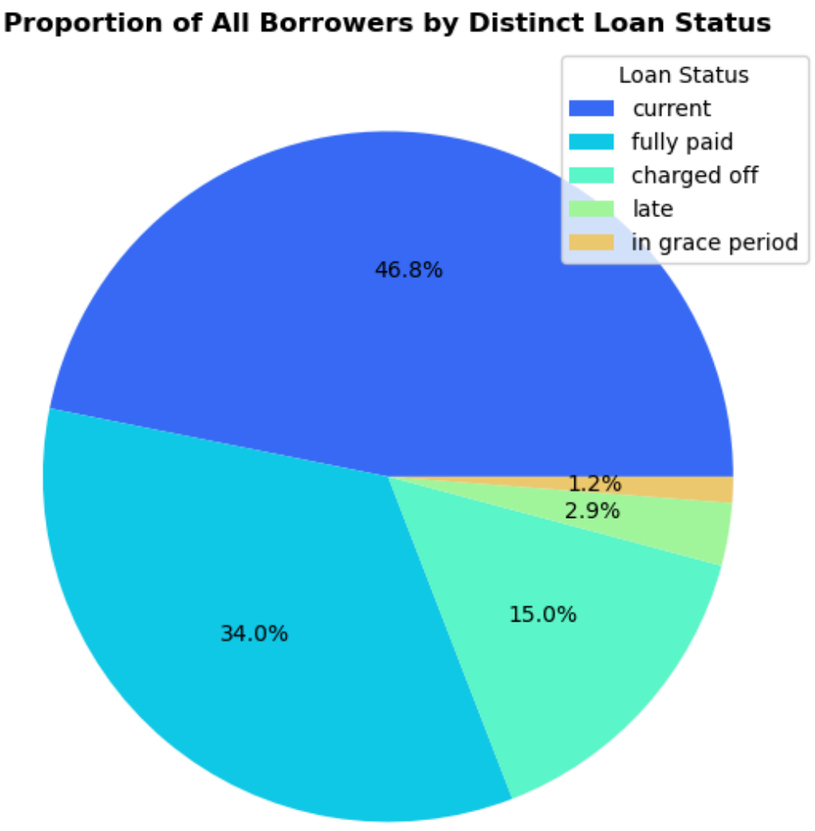
Check total borrowers by distinct loan_status values

Most borrowers are current on their loans (80%), while fully paid loans make up nearly 58% of all borrowers, indicating a strong repayment trend overall. However, a notable 25% of borrowers have charged-off loans, highlighting a substantial risk segment. Smaller portions of loans are either late (5%) or in grace period (2%), signaling pockets where monitoring and proactive collection strategies could mitigate potential defaults. The bar and pie plots clearly illustrate the distribution and relative scale of these repayment statuses.

In [25]:
```
query = """SELECT loan_status, COUNT(DISTINCT(borrower_id)) AS 'total_count',
            COUNT(DISTINCT(borrower_id)) * 100/ (SELECT COUNT(DISTINCT(borrower_id)) FROM loans) AS 'percent_borrowers'
            FROM loans
            GROUP BY loan_status
            ORDER BY total_count DESC;
            """

df = run_query(conn,query)
print(df,end = '\n\n')

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'loan_status', y = 'total_count',palette = 'inferno',ax = ax[0])
ax[0].legend(title="Loan Status")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Loan Status',fontweight = 'bold')

ax[1].pie(df['percent_borrowers'],autopct = '%1.1f%%',colors = sns.color_palette('rainbow'))
ax[1].legend(labels = df['loan_status'],title = "Loan Status")
ax[1].set_title('Proportion of All Borrowers by Distinct Loan Status',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
      loan_status  total_count percent_borrowers
0         current        39792           79.6637
1      fully paid        28919           57.8959
2     charged off        12729           25.4835
3            late         2494            4.9930
4  in grace period         1034            2.0701
```





In [ ]:

KEY AREAS OF ANALYSIS

A. Borrower Profile Analysis- "What kind of customers are we lending to?"

Before lending money, banks must understand who their customers are. This section builds a borrower profile to reveal their income, employment, home ownership, and financial strength.

Summary-

The borrower base is largely middle-income, with most earning between $50k–$100k, and concentrated in stable employment (3–9 years). Mortgage holders and verified borrowers tend to have higher incomes, though a notable subset of high-income renters exists, highlighting exceptions to traditional ownership-income patterns. Repeat borrowing is common, with borrowers often holding multiple loans and high total accounts, indicating potential exposure to over-leverage. Employment length and verification status together reveal that longer-tenured and verified borrowers consistently earn more, while very few high-income borrowers carry excessive debt (DTI > 43%). Overall, the portfolio shows a mix of moderate-to-high financial activity, underlining the importance of income verification, employment stability, and careful monitoring of repeat borrowers for managing lending risk.

In [ ]:

1. Total Borrowers by Number of Loans. Identifies repeat borrowers and potential over-leverage.

The borrower base is remarkably evenly distributed across the number of loans, with roughly 25% of borrowers having 1, 2, 3, or 4 loans. This indicates a balanced mix of single-loan and repeat borrowers. From a risk perspective, while repeat borrowers may bring experience and loyalty, they could also signal potential over-leverage, which the bank should monitor carefully. Overall, the lending portfolio does not appear heavily concentrated in any single borrower segment based on the number of loans.

In [26]:
```
query = """SELECT num_loans,COUNT(borrower_id) AS 'num_borrowers',
           COUNT(borrower_id) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_borrowers'
           FROM (SELECT borrower_id, COUNT(loan_id) AS 'num_loans' FROM loans
                 GROUP BY borrower_id
                 ORDER BY num_loans DESC) t
           GROUP BY num_loans
           ORDER BY num_loans DESC;"""

df = run_query(conn,query)
print(df)
```

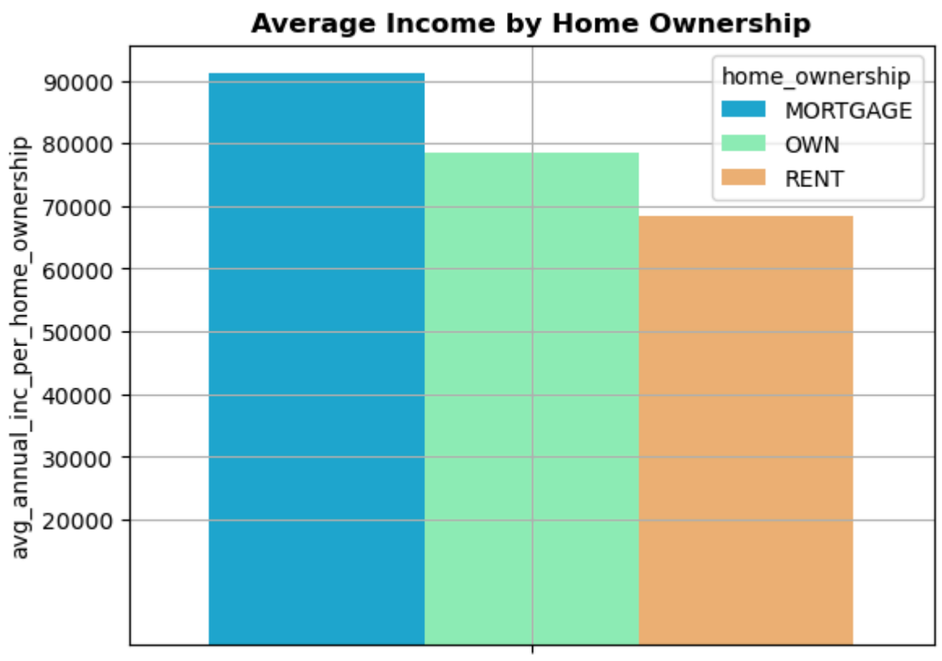
```
   num_loans  num_borrowers percent_borrowers
0          4          12567           25.1592
1          3          12404           24.8328
2          2          12416           24.8569
3          1          12563           25.1512
```

2. Average Income by Home Ownership. Helps check whether owning or renting affects income levels.

Income levels vary noticeably across home ownership types. Borrowers with a mortgage have the highest average annual income ($91k), followed by those who own their homes outright ($78.5k), while renters earn the least on average (~ $68.5k). This suggests that home ownership correlates with financial capacity, and lending strategies could factor in ownership status as a proxy for income strength and repayment ability.

In [27]:
```
query = """SELECT home_ownership,
           ROUND(AVG(annual_inc) ,2) AS 'avg_annual_inc_per_home_ownership'
           FROM borrowers
           GROUP BY home_ownership
           ORDER BY avg_annual_inc_per_home_ownership DESC;"""

df = run_query(conn,query)
print(df)

sns.barplot(data = df,hue = 'home_ownership', y = 'avg_annual_inc_per_home_ownership',palette = 'rainbow')
plt.grid(True)
plt.title('Average Income by Home Ownership',fontweight = 'bold')
plt.yticks(ticks = range(20000,90001,10000))
plt.show()
```

```
  home_ownership  avg_annual_inc_per_home_ownership
0       MORTGAGE                          91152.28
1            OWN                          78574.85
2           RENT                          68516.89
```



3. Borrowers with High Total Accounts. Borrowers with total_acc > 25 are often considered higher credit risk, since too many credit accounts can lead to overextension, difficulty tracking payments, and increased probability of delinquency.

A significant portion of borrowers—about 20,513 individuals, or 41% of the total—have more than 25 credit accounts. This indicates a substantial segment of borrowers who may be over-leveraged, potentially facing difficulty managing multiple credit lines. Banks should monitor these borrowers closely, as high total accounts can increase the likelihood of late payments or defaults.

In [28]:
```
query = """SELECT COUNT(*) AS 'total_borrowers' , COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_borrowers'
           FROM(SELECT borrower_id,total_acc FROM borrowers
                  WHERE total_acc > 25
                  ORDER BY total_acc DESC) t;"""

df = run_query(conn,query)
print(df,end='\n\n')
print('Total Borrowers with Hight Total Accounts are',int(df['total_borrowers'].values[0]),'which is',round(df['percent_borrowers'].values[0],2),
      'of total borrowers.')
```

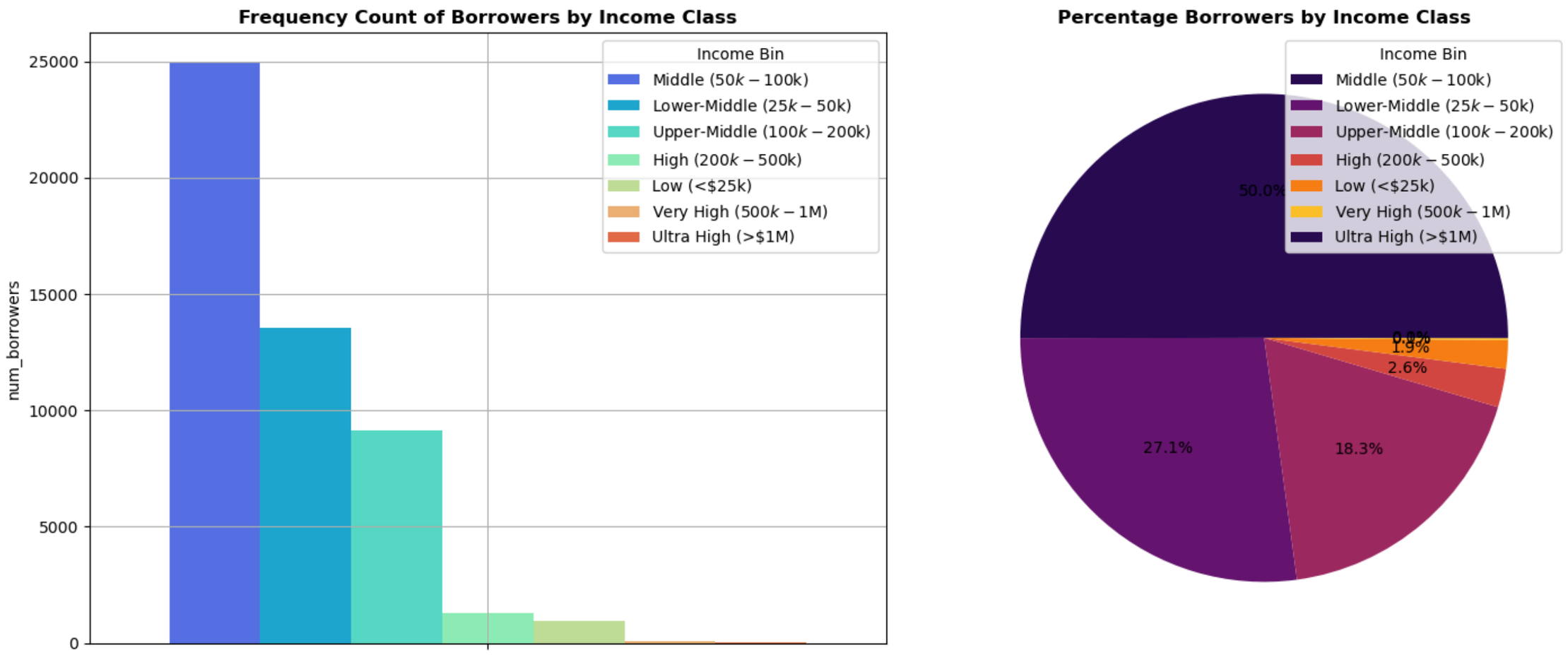
```
   total_borrowers percent_borrowers
0            20513           41.0671

Total Borrowers with Hight Total Accounts are 20513 which is 41.07 of total borrowers.
```

4. Borrower Income Brackets. Gives a real-world income segmentation of the borrower base, showing which class dominates the portfolio.

The majority of borrowers fall into the Middle ($50k–$100k) income bracket, making up 50% of the portfolio, followed by Lower-Middle ($25k–$50k) at 27%. High-income borrowers (>$200k) represent only a small fraction, less than 3% combined. This shows that the lending portfolio is primarily composed of middle-income individuals, highlighting the bank’s focus on mainstream borrowers rather than ultra-wealthy clients.

In [29]:
```
query = """SELECT t.income_bin,COUNT(*) AS 'num_borrowers', COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_borrowers'
           FROM
               (SELECT annual_inc,
                 CASE
                   WHEN annual_inc < 25000 THEN 'Low (<$25k)'
                   WHEN annual_inc BETWEEN 25000 AND 50000 THEN 'Lower-Middle ($25k-$50k)'
                   WHEN annual_inc BETWEEN 50001 AND 100000 THEN 'Middle ($50k-$100k)'
                   WHEN annual_inc BETWEEN 100001 AND 200000 THEN 'Upper-Middle ($100k-$200k)'
                   WHEN annual_inc BETWEEN 200001 AND 500000 THEN 'High ($200k-$500k)'
                   WHEN annual_inc BETWEEN 500001 AND 1000000 THEN 'Very High ($500k-$1M)'
                   ELSE 'Ultra High (>$1M)'
                 END AS 'income_bin'
               FROM borrowers)  t
           GROUP BY t.income_bin
           ORDER BY num_borrowers DESC;"""

df = run_query(conn,query)
print(df)

fig,ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

sns.barplot(data = df,hue = 'income_bin', y = 'num_borrowers',palette = 'rainbow',ax = ax[0])
ax[0].legend(title="Income Bin")
ax[0].grid(True)
ax[0].set_title('Frequency Count of Borrowers by Income Class',fontweight = 'bold')

ax[1].pie(df['percent_borrowers'],autopct = '%1.1f%%',colors = sns.color_palette('inferno'))
ax[1].legend(labels = df['income_bin'],title = "Income Bin",loc = 'best')
ax[1].set_title('Percentage Borrowers by Income Class',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
                 income_bin  num_borrowers  percent_borrowers
0        Middle ($50k-$100k)          24983            50.0160
1    Lower-Middle ($25k-$50k)          13552            27.1311
2  Upper-Middle ($100k-$200k)           9127            18.2723
3          High ($200k-$500k)           1274             2.5506
4                Low (<$25k)            940             1.8819
5       Very High ($500k-$1M)             66             0.1321
6           Ultra High (>$1M)              8             0.0160
```



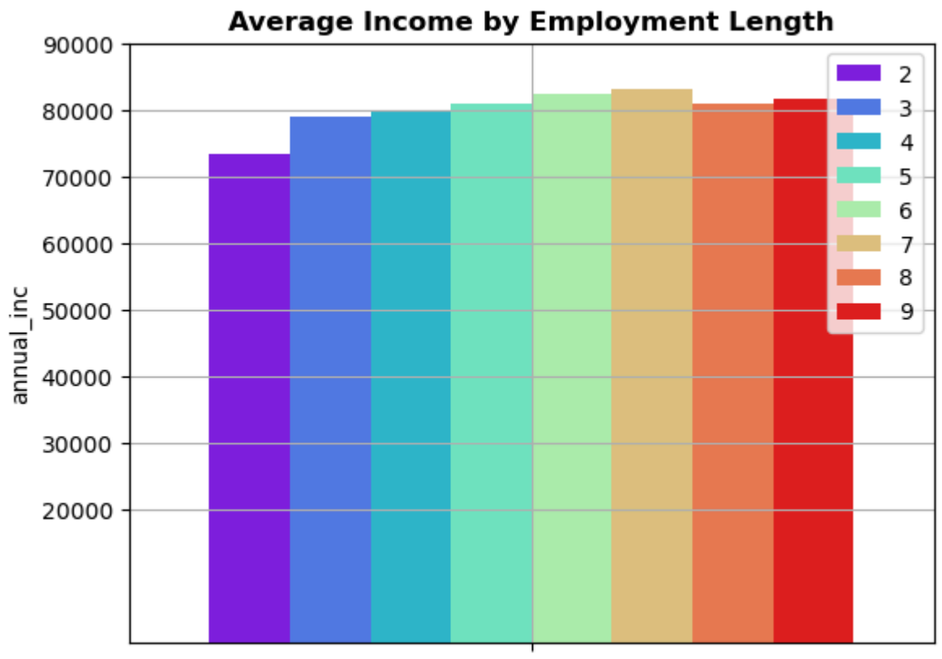
5. Employment Length Segmentation. Longer employment usually reflects stability; useful for assessing repayment capacity.

Borrowers with longer employment histories generally earn higher incomes, with those employed 7–9 years averaging around $82–83k annually. Shorter-tenured employees, particularly those with 2 years of experience, earn noticeably less, around $73k. This indicates that employment stability correlates with higher earning capacity, helping lenders gauge repayment reliability.

In [30]:
```
query = """SELECT emp_length, ROUND(AVG(annual_inc),2) AS 'avg_annual_income_in$'
           FROM borrowers
           GROUP BY emp_length
           ORDER BY emp_length DESC;"""

df = run_query(conn,query)
print(df)
```

```
   emp_length  avg_annual_income_in$
0           9               81550.31
1           8               80988.97
2           7               82956.86
3           6               82343.89
4           5               80805.99
5           4               79717.17
6           3               78867.15
7           2               73377.03
```

In [31]:
```
query = """SELECT emp_length,annual_inc
           FROM borrowers;"""

df = run_query(conn,query)

sns.barplot(data = df,hue = 'emp_length', y = 'annual_inc',palette = 'rainbow',estimator = np.mean,errorbar = None,legend = False)
plt.legend(['2','3','4','5','6','7','8','9'])
plt.grid(True)
plt.title('Average Income by Employment Length',fontweight = 'bold')
plt.yticks(ticks = range(20000,90001,10000))
plt.show()
```

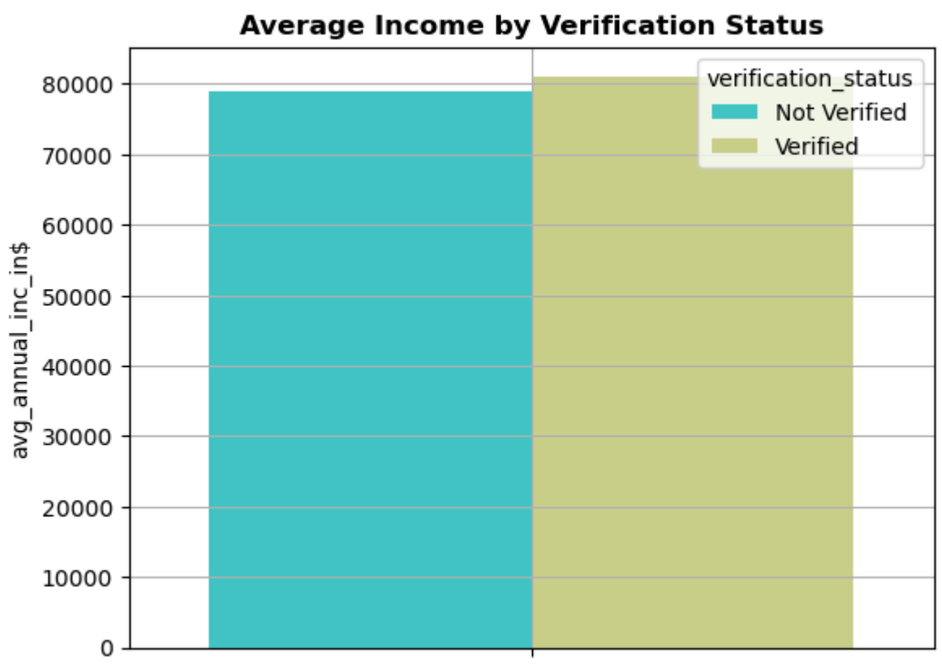


6. Verification Status vs Income. Tests whether verified borrowers report different income levels than non-verified ones.

Verified borrowers report slightly higher average incomes ($81k) compared to non-verified ones ($79k). This suggests that income verification may be associated with higher-earning borrowers, highlighting that verification could serve as a signal of borrower financial strength and potential repayment capacity.

In [32]:
```
query = """SELECT verification_status, ROUND(AVG(annual_inc),2) AS 'avg_annual_inc_in$'
           FROM borrowers
           GROUP BY verification_status
           ORDER BY avg_annual_inc_in$;"""

df = run_query(conn,query)
print(df)

sns.barplot(data = df,hue = 'verification_status', y = 'avg_annual_inc_in$',palette = 'rainbow')
plt.grid(True)
plt.title('Average Income by Verification Status',fontweight = 'bold')
plt.show()
```

```
  verification_status  avg_annual_inc_in$
0        Not Verified            79022.29
1            Verified            81144.58
```



7. Renters with High Income. Some people who earn a lot of money(>100K) are still living in rented houses. This is unusual, and lenders might want to look into why this happens because it doesn't follow the usual pattern of more income -> more likely to own a home.

Around 2,777 borrowers, roughly 6% of the total, are high-income earners (> $100k) who still rent their homes. This is an unusual pattern since higher income typically correlates with homeownership. Lenders may want to investigate these borrowers further, as their behavior could signal unique financial strategies or potential risks.

In [33]:
```
query = """SELECT COUNT(*) AS 'Renters with High Income',
           COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_renters_with_high_inc'
           FROM borrowers
           WHERE home_ownership = 'RENT' AND annual_inc > 100000;"""

df = run_query(conn,query)
print(df,end = '\n\n')
print('Renters with high income are',df['Renters with High Income'].values[0],'which is',
      round(df['percent_renters_with_high_inc'].values[0]),'percent of total borrowers.')
```

```
  Renters with High Income  percent_renters_with_high_inc
0                     2777                         5.5596

Renters with high income are 2777 which is 6 percent of total borrowers.
```

8. High Income + High DTI Borrowers [High risk: > 43% (many lenders use 43% as a cutoff)]. Even wealthy borrowers can be risky if they carry too much debt.

Only 23 borrowers, representing a tiny 0.046% of the portfolio, combine high income with a dangerously high debt-to-income ratio (>43%). While wealthy, these borrowers carry significant debt, making them a concentrated high-risk group that lenders should monitor closely despite their small numbers.

In [34]:
```
query = """SELECT COUNT(*) AS 'num_borrowers', COUNT(*) * 100 / (SELECT COUNT(borrower_id) FROM borrowers) AS 'percent_borrowers'
           FROM loans t1
           JOIN borrowers t2
           ON t1.borrower_id = t2.borrower_id
           WHERE t1.dti > 43  AND t2.annual_inc > 200000;"""

df = run_query(conn,query)
print('The borrowers with High Income + High DTI are',df['num_borrowers'].values[0],'which is',df['percent_borrowers'].values[0],
      'percent of total borrowers')
```

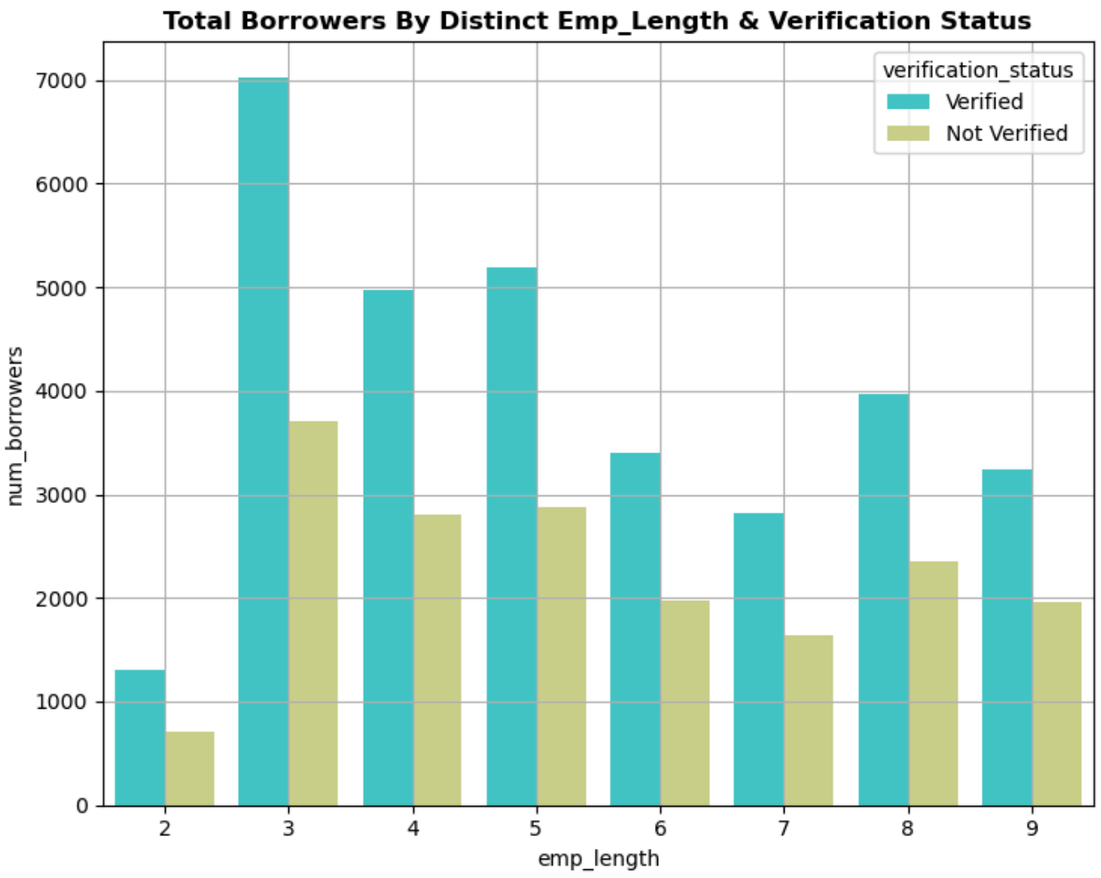
```
The borrowers with High Income + High DTI are 23 which is 0.0460 percent of total borrowers
```

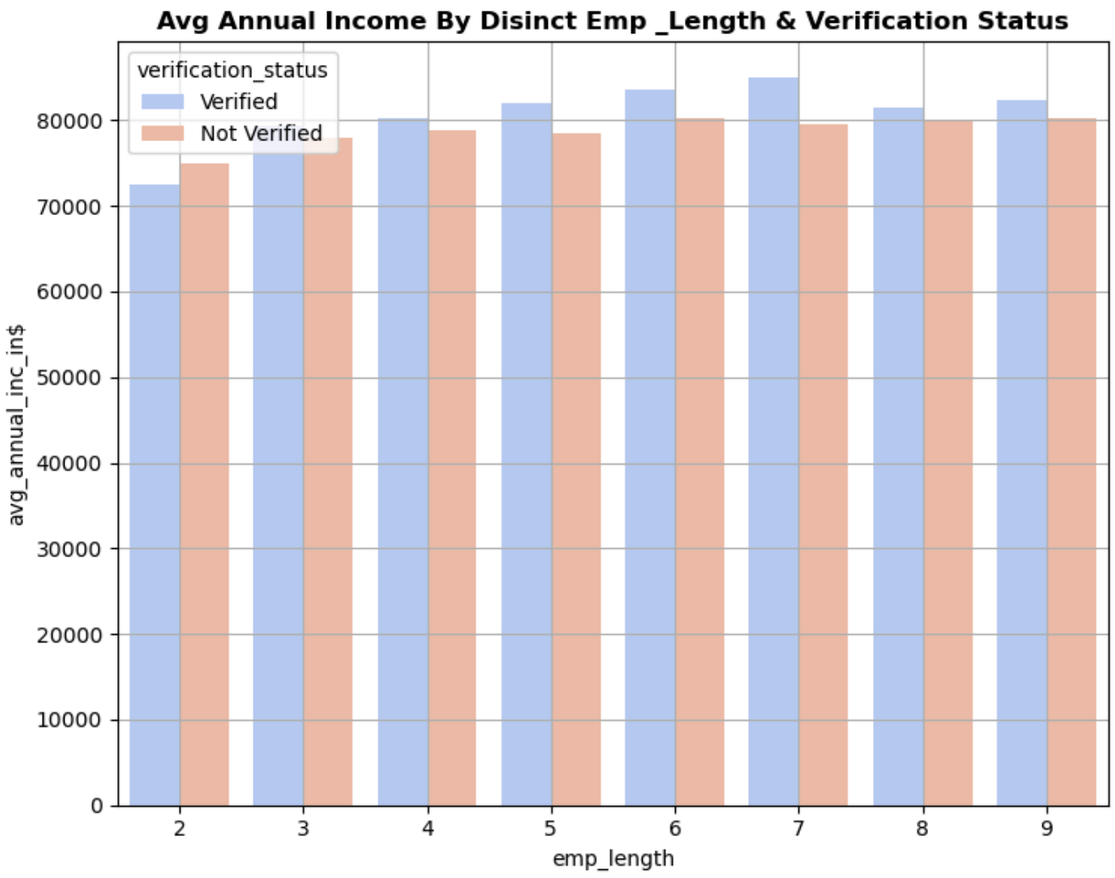
9. Average Annual Income By Employment Length + Verification Segmentation. Reveals combined effect of work history and verified income on borrower profiles.

Borrowers with longer employment histories tend to report higher incomes, and verified borrowers consistently earn more than non-verified ones across all employment lengths. The difference is most pronounced among mid- to long-tenure employees (5–7 years), highlighting that income verification not only confirms reported earnings but also correlates with stronger, more stable borrower profiles.

In [35]:
```
query = """SELECT emp_length,verification_status,COUNT(*) AS 'num_borrowers',
            ROUND(AVG(annual_inc),2) AS 'avg_annual_inc_in$'
            FROM borrowers
            GROUP BY emp_length,verification_status
            ORDER BY emp_length DESC,num_borrowers DESC;"""

df = run_query(conn,query)
print(df)

fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))

sns.barplot(data = df,x = 'emp_length',y = 'num_borrowers',hue = 'verification_status',palette ='rainbow',ax = ax[0])
ax[0].grid(True)
ax[0].set_title('Total Borrowers By Distinct Emp_Length & Verification Status',fontweight = 'bold')

sns.barplot(data = df,x = 'emp_length',y = 'avg_annual_inc_in$',hue = 'verification_status',palette = 'coolwarm',ax = ax[1])
ax[1].grid(True)
ax[1].set_title('Avg Annual Income By Disinct Emp _Length & Verification Status',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
    emp_length verification_status  num_borrowers  avg_annual_inc_in$
0            9            Verified           3240            82304.47
1            9        Not Verified           1959            80302.99
2            8            Verified           3974            81493.30
3            8        Not Verified           2354            80137.56
4            7            Verified           2820            84933.66
5            7        Not Verified           1643            79563.94
6            6            Verified           3398            83524.77
7            6        Not Verified           1976            80313.21
8            5            Verified           5189            82064.63
9            5        Not Verified           2870            78530.34
10           4            Verified           4972            80207.40
11           4        Not Verified           2810            78849.75
12           3            Verified           7020            79332.64
13           3        Not Verified           3712            77986.84
14           2            Verified           1311            72516.42
15           2        Not Verified            702            74984.23
```



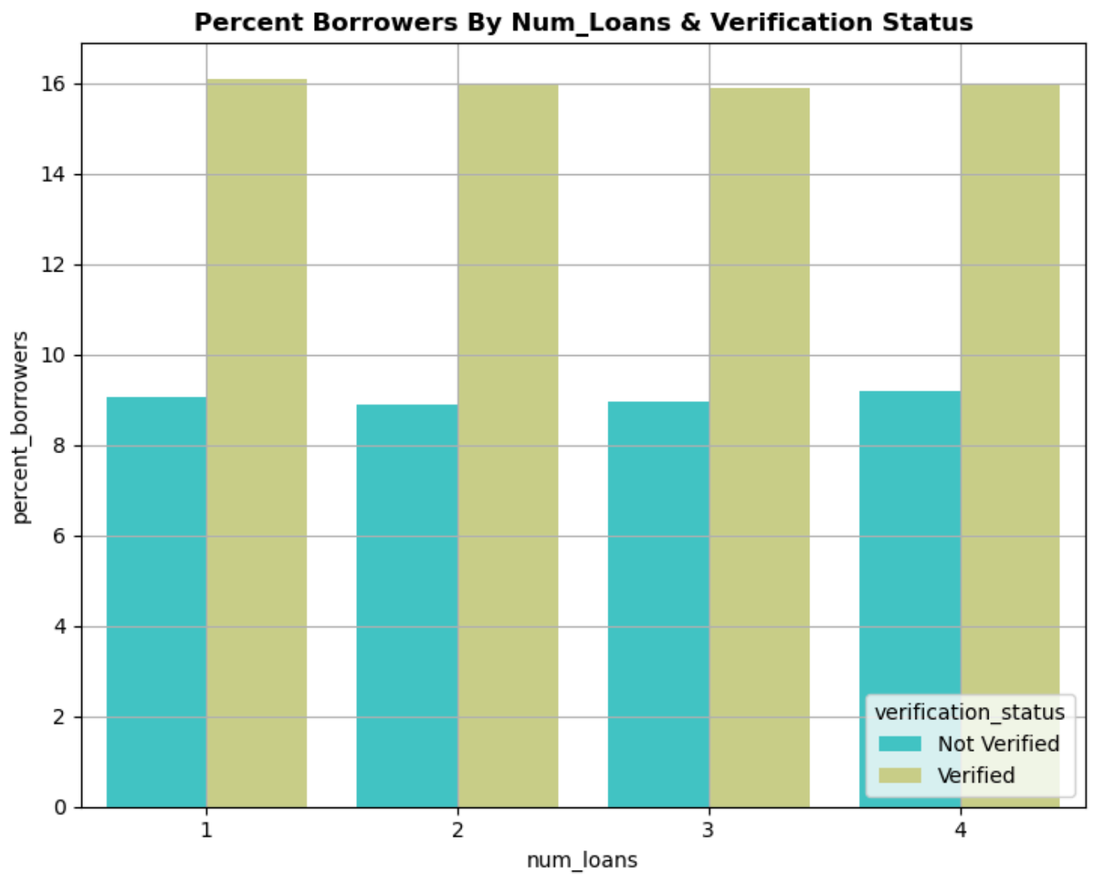


10. Average Annual Income By Number of loans + Verification Status. Exploring How Annual Income & Verification Status Influences the Number of Loans Taken

Verified borrowers not only tend to earn slightly higher incomes than non-verified ones but also dominate across all categories of loan counts. Interestingly, borrowers with multiple loans (2–4) are predominantly verified and represent a larger share of the portfolio, suggesting that lenders rely more on verified income reports when approving repeat loans. This reinforces the role of verification in both risk assessment and portfolio management.

In [36]:
```
query = """SELECT t1.num_loans,t2.verification_status,COUNT(t2.borrower_id) AS 'num_borrowers',
            ROUND(COUNT(t2.borrower_id) * 100 / (SELECT COUNT(borrower_id) FROM borrowers),2) AS 'percent_borrowers',
            ROUND(AVG(t2.annual_inc),2) AS 'avg_annual_inc_in$'
            FROM (SELECT borrower_id, COUNT(loan_id) AS 'num_loans' FROM loans
                  GROUP BY borrower_id
                  ORDER BY num_loans DESC) t1
            JOIN borrowers t2
            ON t1.borrower_id = t2.borrower_id
            GROUP BY t1.num_loans,t2.verification_status
            ORDER BY t1.num_loans DESC;"""

df = run_query(conn,query)
print(df)

fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))

sns.barplot(data = df,x = 'num_loans', y = 'avg_annual_inc_in$',hue = 'verification_status',palette = 'coolwarm',ax = ax[0])
ax[0].set_title('Avg Annual Income By Num_Loans & Verification Status',fontweight = 'bold')
ax[0].legend(title = 'verification_status',loc = 'lower right')
ax[0].grid(True)

sns.barplot(data = df,x = 'num_loans',y = 'percent_borrowers',hue = 'verification_status',palette = 'rainbow',ax = ax[1])
ax[1].set_title('Percent Borrowers By Num_Loans & Verification Status',fontweight = 'bold')
ax[1].legend(title = 'verification_status',loc = 'lower right')
ax[1].grid(True)

plt.tight_layout()
plt.show()
```

```
   num_loans verification_status  num_borrowers percent_borrowers  \
0          4        Not Verified           4583              9.18
1          4            Verified           7984             15.98
2          3        Not Verified           4473              8.95
3          3            Verified           7931             15.88
4          2        Not Verified           4438              8.88
5          2            Verified           7978             15.97
6          1        Not Verified           4532              9.07
7          1            Verified           8031             16.08

   avg_annual_inc_in$
0            79026.97
1            80774.95
2            78650.15
3            82495.77
4            78720.55
5            80303.08
6            79680.34
7            81013.64
```



In [ ]:

In [ ]:

B. Loan Portfolio Overview- "What does our loan portfolio look like?"

Every financial institution must know where its money is going. Here, we analyze the portfolio of loans issued, studying their amounts, terms, grades, and purposes to see how the banks allocates credit.

Summary-

The loan portfolio overview reveals that most lending is concentrated in medium-term (36 months) loans, with average amounts around $13–21K depending on the term. Higher-risk grades (D–G) tend to have higher interest rates and slightly larger loans, while safer grades (A–C) dominate the number of loans issued. Debt consolidation and credit card purposes are the largest drivers of the portfolio, both in number and total loan amount, whereas niche purposes like small business, home improvement, and renewable energy contribute minimally. High-risk borrowers often have high revolving utilization or costly loans, as seen in the top-ranked loans with high interest rates and DTI, highlighting potential default risks. Combined analyses by term, grade, and purpose show that longer-term and higher-purpose loans carry higher rates, and the grade-purpose stack plots indicate moderate diversification with heavy concentration in a few key segments. The plots effectively visualize distributions, comparisons, and portfolio concentration, helping identify where credit exposure is highest.

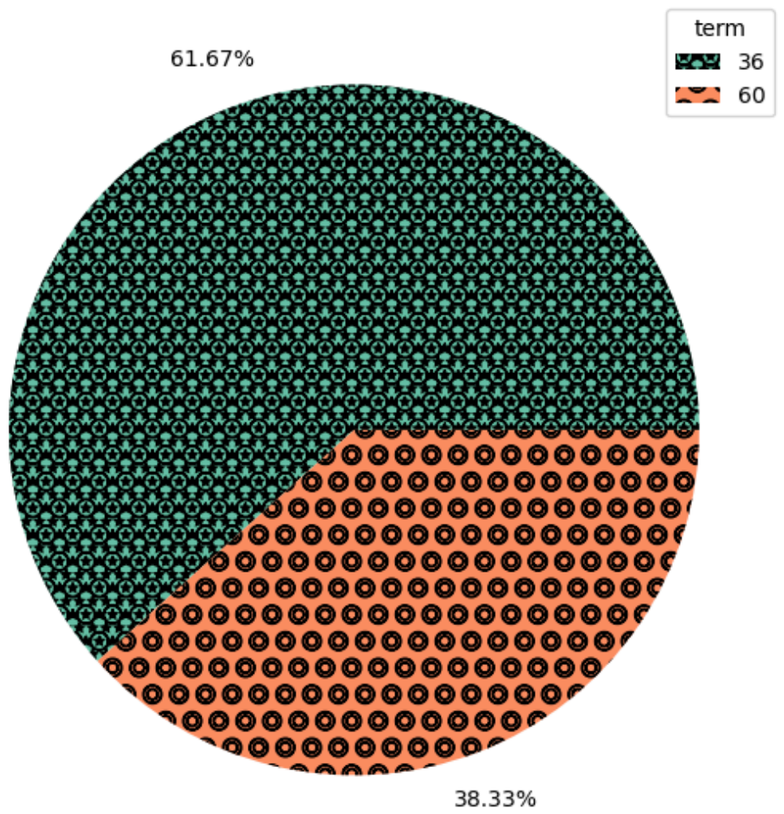
In [ ]:

11. Loan Amounts by Term. Helps compare lending strategies in short-term vs long-term loans.

The analysis shows a clear distinction between short-term (36 months) and long-term (60 months) loans. While short-term loans dominate the portfolio by number of loans, averaging around $13.7k, they account for roughly 62% of total loans issued. In contrast, long-term loans are fewer but larger, averaging $20.9k, contributing 38% to the portfolio. This indicates the bank balances between frequent smaller loans and less frequent larger commitments, diversifying risk and catering to different borrower needs.

In [37]:
```
query = """SELECT term, ROUND(AVG(loan_amnt),2) AS 'avg_loan_amnt_per_term',
           ROUND(SUM(loan_amnt) * 100 / (SELECT SUM(loan_amnt) FROM loans),2) AS 'percent_loan_amnt'
           FROM loans
           GROUP BY term
           ORDER BY term;"""

df = run_query(conn,query)
print(df)

fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))

sns.barplot(data = df,hue = 'term',y = 'avg_loan_amnt_per_term',palette = 'rainbow',ax = ax[0])
ax[0].grid(True)
ax[0].set_title('Avg Loan Amount By Term',fontweight = 'bold')

ax[1].pie(df['percent_loan_amnt'],autopct = '%1.2f%%',hatch = ['**O', 'oO'],pctdistance = 1.15,colors = sns.color_palette('Set2'))
ax[1].legend(labels = df['term'],loc = 'best',title = 'term')
ax[1].set_title('Percent Loan Amount by Term',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
  term avg_loan_amnt_per_term percent_loan_amnt
0   36               13735.59             61.67
1   60               20945.90             38.33
```

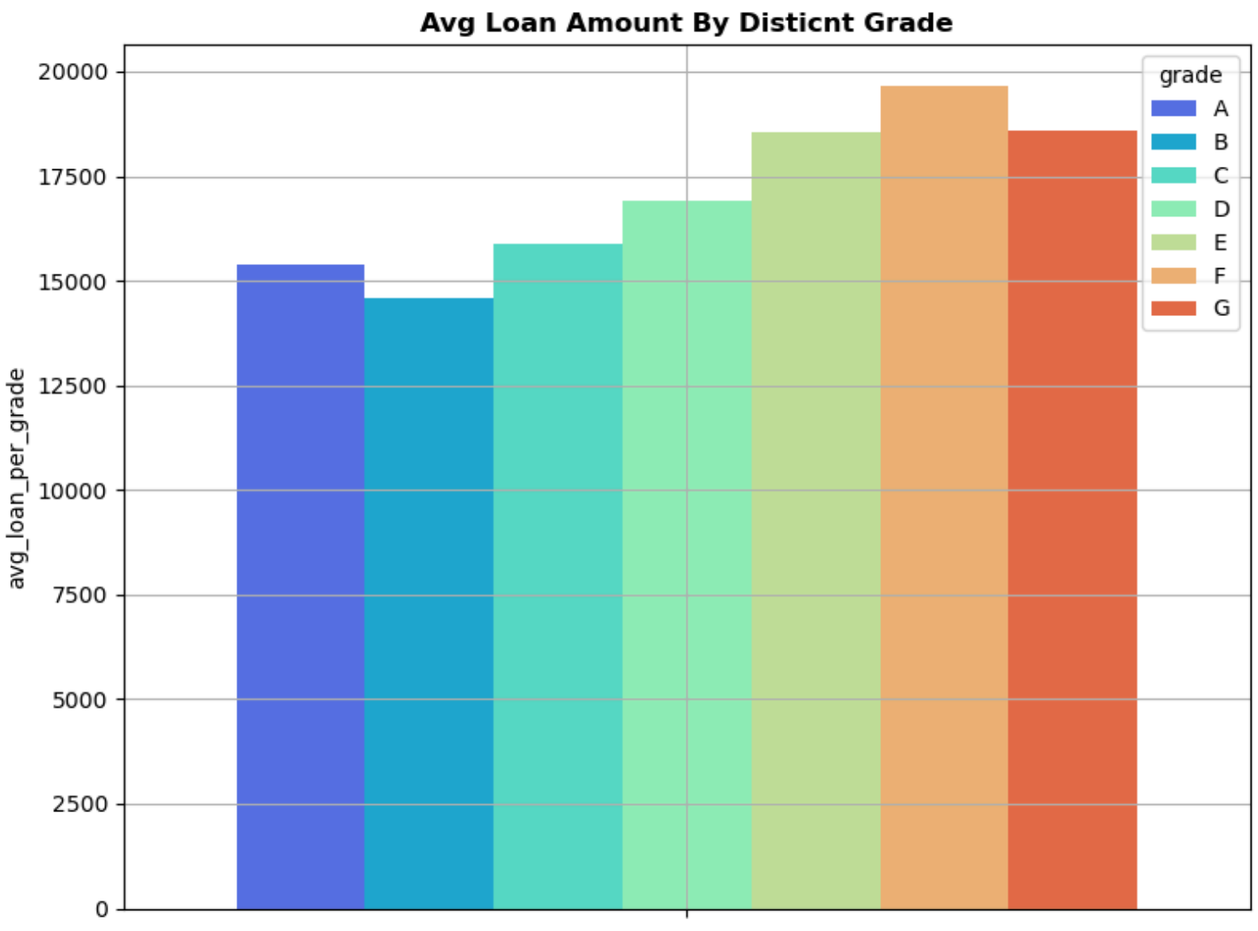


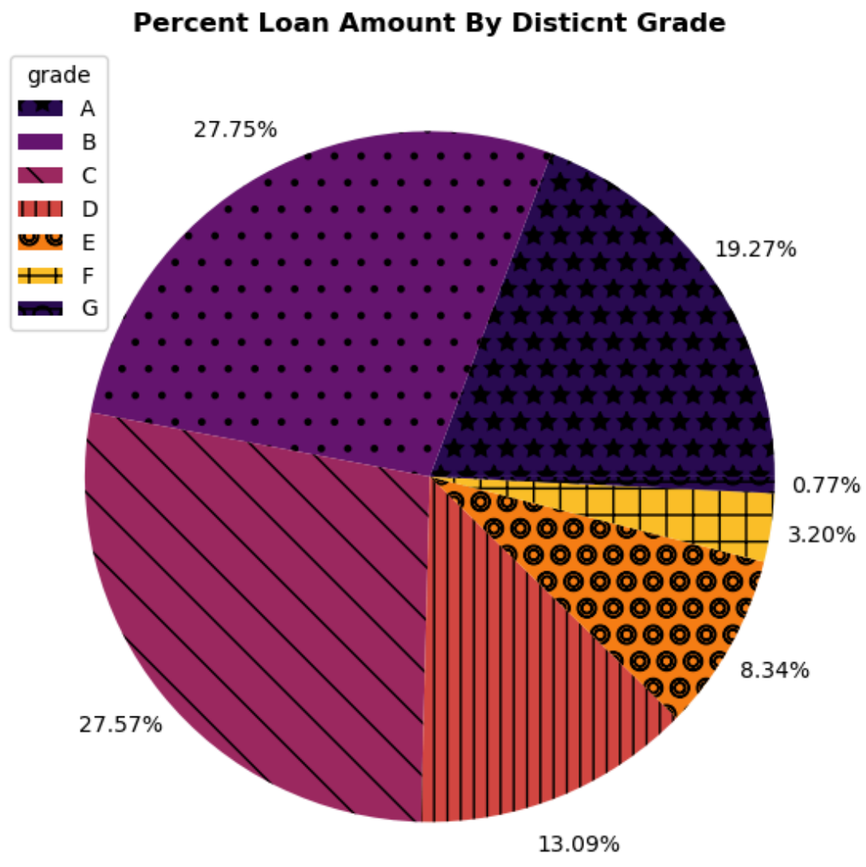
12. Loan Amounts by Grade. Assesses whether higher credit grades qualify for larger loans.

The portfolio shows that mid-range grades (B and C) dominate the total loan share, together accounting for over 55% of the portfolio, even though their average loan amounts ($14.6k–$15.9k) are moderate. Higher-risk grades (E, F, G) receive fewer loans but at higher average amounts ($18.5k–$19.6k), suggesting the bank extends larger loans to riskier borrowers, likely for higher interest returns. Conversely, top-grade borrowers (A) get moderate loan amounts but still represent a significant 19% of the portfolio, reflecting conservative lending to low-risk clients.

In [38]:
```
query = """SELECT grade, ROUND(AVG(loan_amnt),2) AS 'avg_loan_per_grade',
           ROUND(SUM(loan_amnt) * 100 / (SELECT SUM(loan_amnt) FROM loans),2) AS 'percent_loan_amnt'
           FROM loans
           GROUP BY grade
           ORDER BY grade;"""

df = run_query(conn,query)
print(df)

fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))

sns.barplot(data = df,hue = 'grade',y = 'avg_loan_per_grade',palette = 'rainbow',ax = ax[0])
ax[0].grid(True)
ax[0].set_title('Avg Loan Amount By Disticnt Grade',fontweight = 'bold')

ax[1].pie(df['percent_loan_amnt'],autopct = '%1.2f%%',hatch = ['*','.','\\','||','oO','+','-O-'],
          pctdistance = 1.15,colors = sns.color_palette('inferno'))
ax[1].legend(labels = df['grade'],loc = 'best',title = 'grade')
ax[1].set_title('Percent Loan Amount By Disticnt Grade',fontweight = 'bold')

plt.tight_layout()
plt.show()
```

```
  grade avg_loan_per_grade percent_loan_amnt
0     A           15398.34             19.27
1     B           14586.26             27.75
2     C           15862.89             27.57
3     D           16927.36             13.09
4     E           18548.09              8.34
5     F           19650.41              3.20
6     G           18597.05              0.77
```





13. Interest Rates and Total Loans by Grade. Confirms whether higher-risk borrowers face higher costs

The analysis confirms that higher-risk borrowers are charged higher interest rates. Top-grade borrowers (A) enjoy the lowest average rate of 6.68%, while the riskiest borrowers (G) face 27.99%. Interestingly, the bulk of loans (grades B and C) account for nearly 58% of total loans, showing the bank balances risk and volume by extending moderate rates to mid-risk clients. This tiered pricing reflects prudent risk-based lending practices while maximizing portfolio exposure.

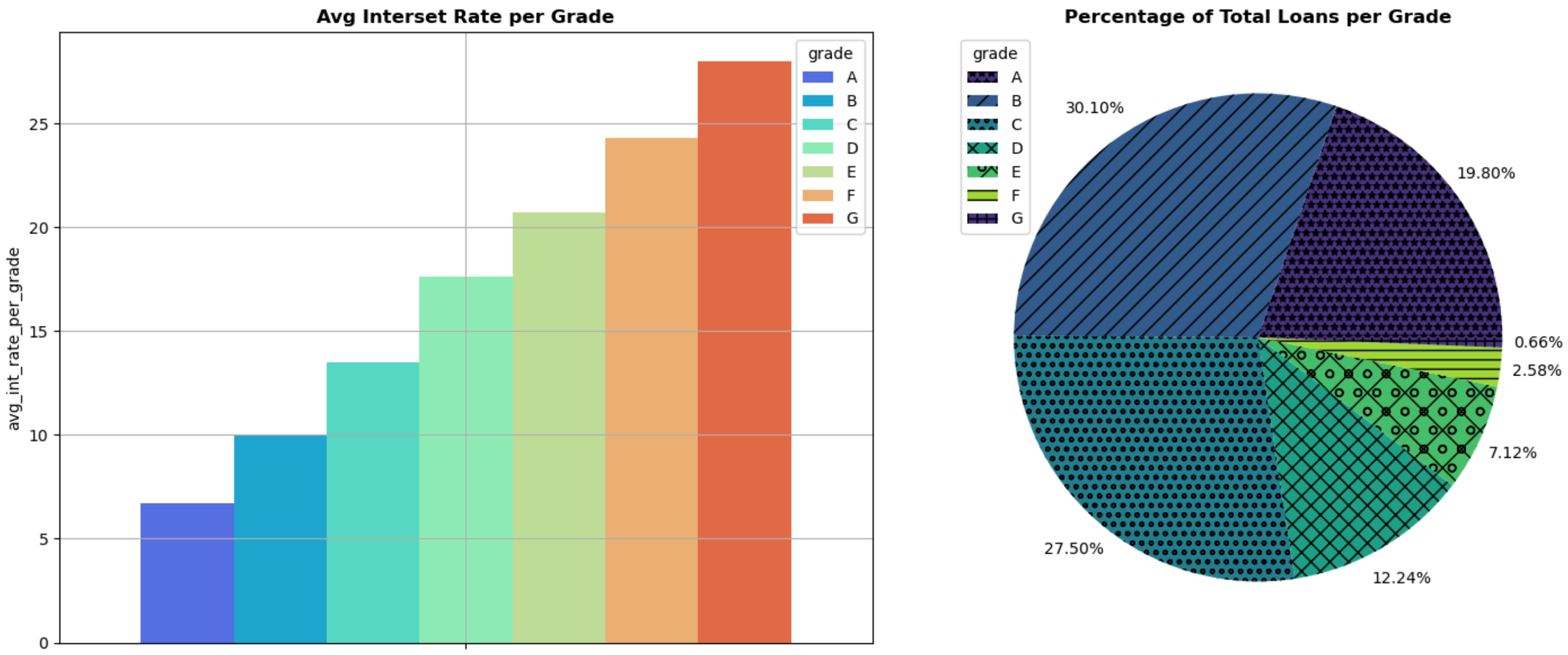
In [39]:
```
query = """SELECT grade, ROUND(AVG(int_rate),2) AS 'avg_int_rate_per_grade',
           COUNT(loan_id) AS 'num_loans',
           ROUND(COUNT(loan_id) * 100 / (SELECT COUNT(loan_id) FROM loans),2) AS 'percent_num_loans'
           FROM loans
           GROUP BY grade
           ORDER BY grade;
           """
```

```
df = run_query(conn,query)
df
```

Out[39]:

	grade	avg_int_rate_per_grade	num_loans	percent_num_loans
0	A	6.68	24725	19.80
1	B	9.97	37598	30.11
2	C	13.47	34347	27.51
3	D	17.60	15279	12.24
4	E	20.74	8885	7.12
5	F	24.29	3220	2.58
6	G	27.99	821	0.66

In [40]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))

sns.barplot(data = df,hue = 'grade',y = 'avg_int_rate_per_grade',palette = 'rainbow',ax = ax[0])
ax[0].grid(True)
ax[0].set_title('Avg Interset Rate per Grade',fontweight = 'bold')

ax[1].pie(df['percent_num_loans'],autopct = '%1.2f%%',hatch = ['**','//','oo','xX','ox','--','|--|'],pctdistance = 1.15,
        colors = sns.color_palette('viridis'))
ax[1].legend(labels = df['grade'],title = 'grade')
ax[1].set_title('Percentage of Total Loans per Grade',fontweight = 'bold')

plt.tight_layout()
plt.show()
```



14. Loan Purpose Ranking. Identifies the biggest demand drivers in the portfolio.

Most loans in the portfolio are for debt consolidation, accounting for 57% of total loans with an average amount of around $16.6K, followed by credit card loans at 24% and home improvement at 6%. Smaller purposes like medical, moving, and vacation together contribute very little. The bar plots clearly show average and total loan amounts per purpose, while the line plot illustrates how average loan amounts vary with the number of loans. The pie chart highlights the share of total loans among the top 5 purposes, making it easy to see which segments dominate. Overall, the portfolio is heavily concentrated in a few key purposes, indicating where lending activity is most focused and where risk exposure may be highest.

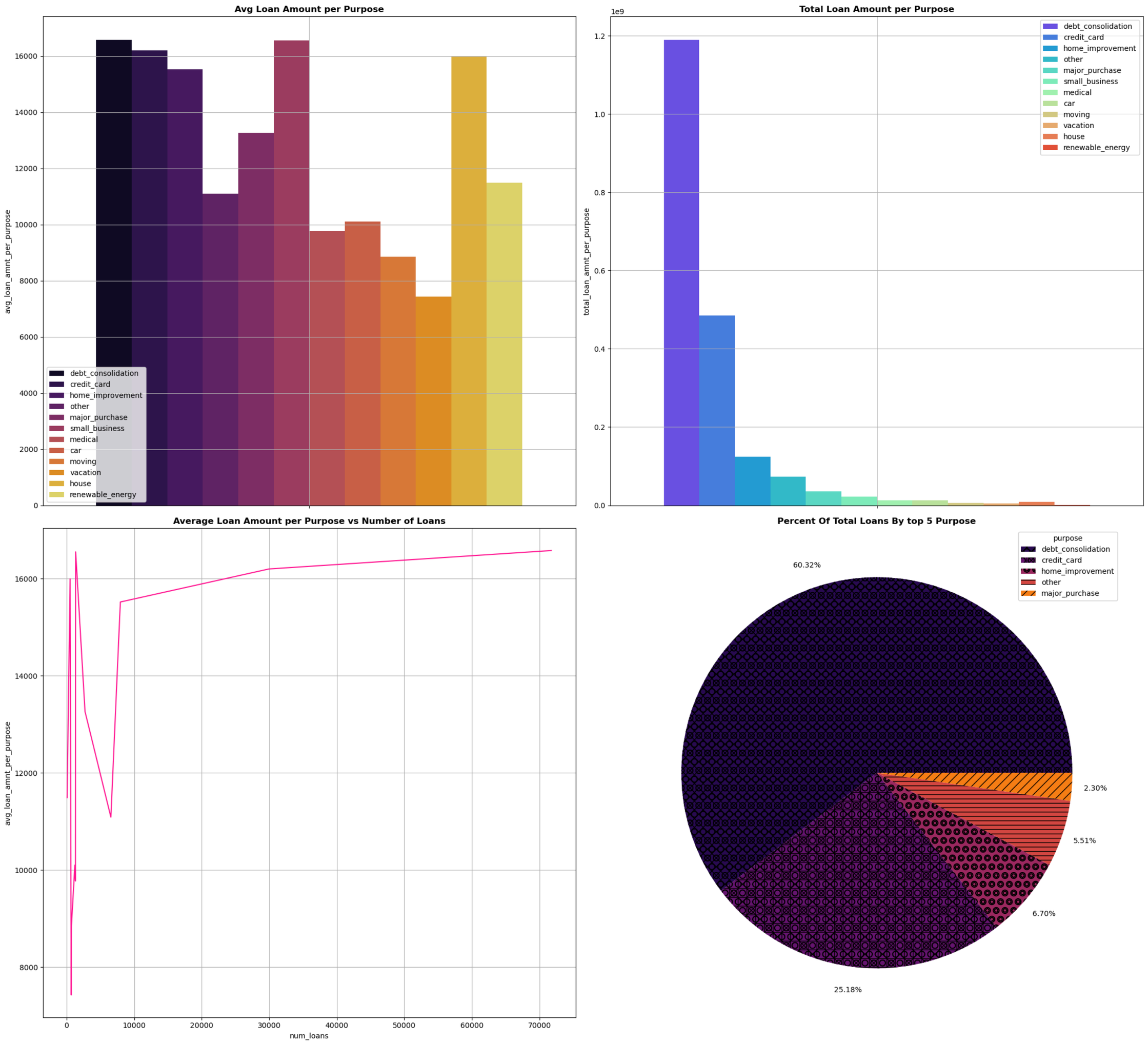
In [41]:
```
query = """SELECT purpose, SUM(loan_amnt) AS 'total_loan_amnt_per_purpose',
            ROUND(AVG(loan_amnt),2) AS 'avg_loan_amnt_per_purpose',
            COUNT(loan_id) AS 'num_loans',
            ROUND(COUNT(loan_id) * 100 / (SELECT COUNT(loan_id) FROM loans),2) AS 'percent_num_loans'
            FROM loans
            GROUP BY purpose
            ORDER BY percent_num_loans DESC;
            """

df = run_query(conn,query)
df
```

Out[41]:

	purpose	total_loan_amnt_per_purpose	avg_loan_amnt_per_purpose	num_loans	percent_num_loans
0	debt_consolidation	1189788350	16579.18	71764	57.47
1	credit_card	485344625	16199.75	29960	23.99
2	home_improvement	123553325	15519.82	7961	6.38
3	other	72741800	11090.38	6559	5.25
4	major_purchase	36246050	13262.37	2733	2.19
5	small_business	22246300	16552.31	1344	1.08
6	medical	12864300	9775.30	1316	1.05
7	car	12375650	10102.57	1225	0.98
8	moving	6375850	8855.35	720	0.58
9	vacation	5111550	7429.58	688	0.55
10	house	8366275	15996.70	523	0.42
11	renewable_energy	942400	11492.68	82	0.07

In [42]:
```
fig,ax = plt.subplots(nrows = 2, ncols = 2,figsize = (22,20))

sns.barplot(data = df,hue = 'purpose',y = 'avg_loan_amnt_per_purpose',palette = 'inferno',ax = ax[0,0])
ax[0,0].grid(True)
ax[0,0].legend(loc = 'lower left')
ax[0,0].set_title('Avg Loan Amount per Purpose',fontweight = 'bold')

sns.barplot(data = df,hue = 'purpose',y = 'total_loan_amnt_per_purpose',palette = 'rainbow',ax = ax[0,1])
ax[0,1].grid(True)
ax[0,1].legend(loc = 'best')
```

```
ax[0,1].set_title('Total Loan Amount per Purpose',fontweight = 'bold')

sns.lineplot(data = df,x = 'num_loans', y = 'avg_loan_amnt_per_purpose',markers = True,ax = ax[1,0],color = 'deeppink')
ax[1,0].grid(True)
ax[1,0].set_title('Average Loan Amount per Purpose vs Number of Loans',fontweight = 'bold')

ax[1,1].pie(df['percent_num_loans'].head(5),autopct = '%1.2f%%',hatch = ['xOX','X+O','oO','--','//'],pctdistance = 1.12,
          colors = sns.color_palette('inferno'))
ax[1,1].legend(labels = df['purpose'],title = 'purpose')
ax[1,1].set_title('Percent Of Total Loans By top 5 Purpose',fontweight = 'bold')

plt.tight_layout()
plt.show()
```



15. Installment-to-Income Ratio. Query to compute (installment / (annual_inc/12)) ratio per borrower. Measures repayment stress relative to income; highlights over-leveraged borrowers.

EMI = [P * r * (1 + r)^n] / [(1 + r)^n - 1]

Where:

- P = Loan Principal
- r = Monthly interest rate = (Annual Interest Rate / 12) / 100
- n = Total number of monthly installments = Loan Tenure (in years) * 12

The installment-to-income analysis highlights the repayment burden across borrowers. From the dataframe, we see extreme cases where monthly installments exceed income multiple times (e.g., borrower_id 16010 has an installment-to-income ratio of 8.41), signaling high stress, though most borrowers maintain a ratio around 2–3. The KDE plots confirm this, showing a concentration of borrowers with moderate installments and ratios, while a few outliers stand out. Scatter plots show that while higher-income borrowers tend to take higher EMIs, their installment-to-income ratio is generally lower, implying more manageable debt. The analysis flags lower-income borrowers with high ratios as the most vulnerable group. The correlation heatmap further supports that income and EMI are related, but the ratio varies independently, emphasizing the need to monitor repayment stress beyond just income levels.

In [43]:
```
query = """SELECT t2.borrower_id,t2.total_monthly_installment,t3.annual_inc,
          ROUND(((t2.total_monthly_installment))/(t3.annual_inc/12),2) AS 'installment_to_income_ratio'
          FROM (SELECT t1.borrower_id, SUM(t1.monthly_installment) AS 'total_monthly_installment'
                  FROM (SELECT borrower_id,
                              ROUND(
                                ((loan_amnt) * ((int_rate/12)/100) * POWER((1 + ((int_rate/12)/100)),term)) /
                                ((POWER((1 + ((int_rate/12)/100)),term)) - 1),2) AS monthly_installment
                         FROM loans) t1
                  GROUP BY t1.borrower_id) t2
                  JOIN borrowers t3
                  ON t2.borrower_id = t3.borrower_id
                  ORDER BY installment_to_income_ratio DESC;"""

# Displaying top 15 over-leveraged borrowers
```

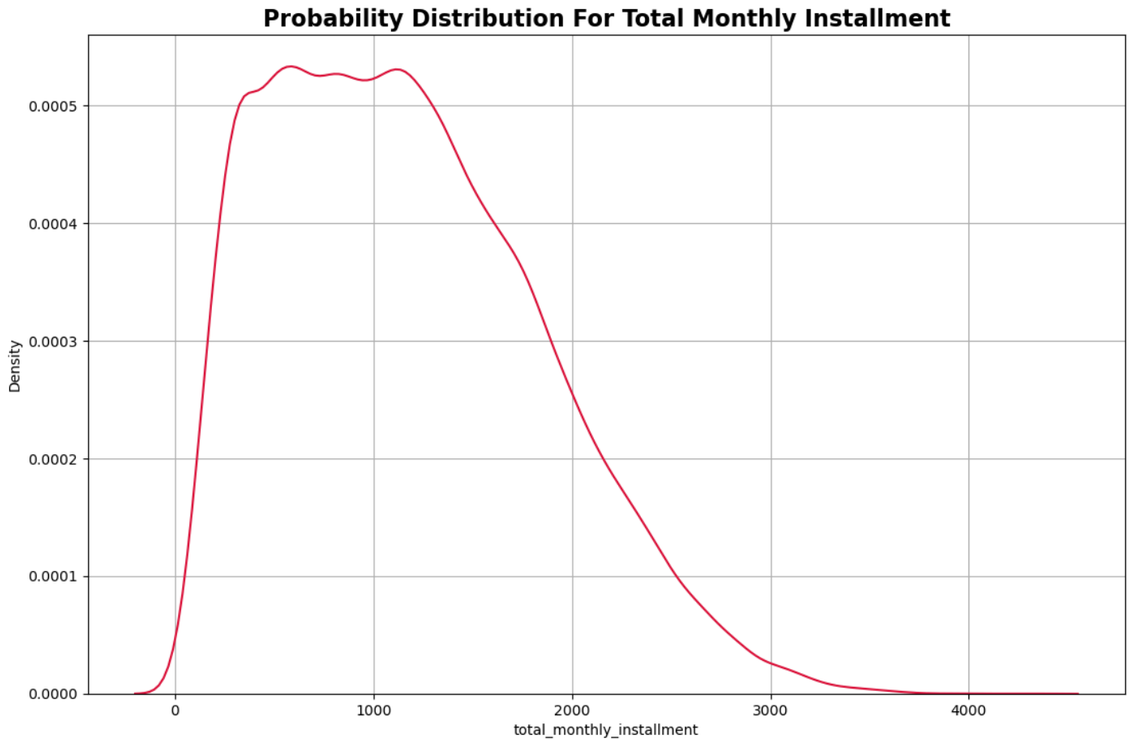
```
df = run_query(conn,query)
df.head(15)
```

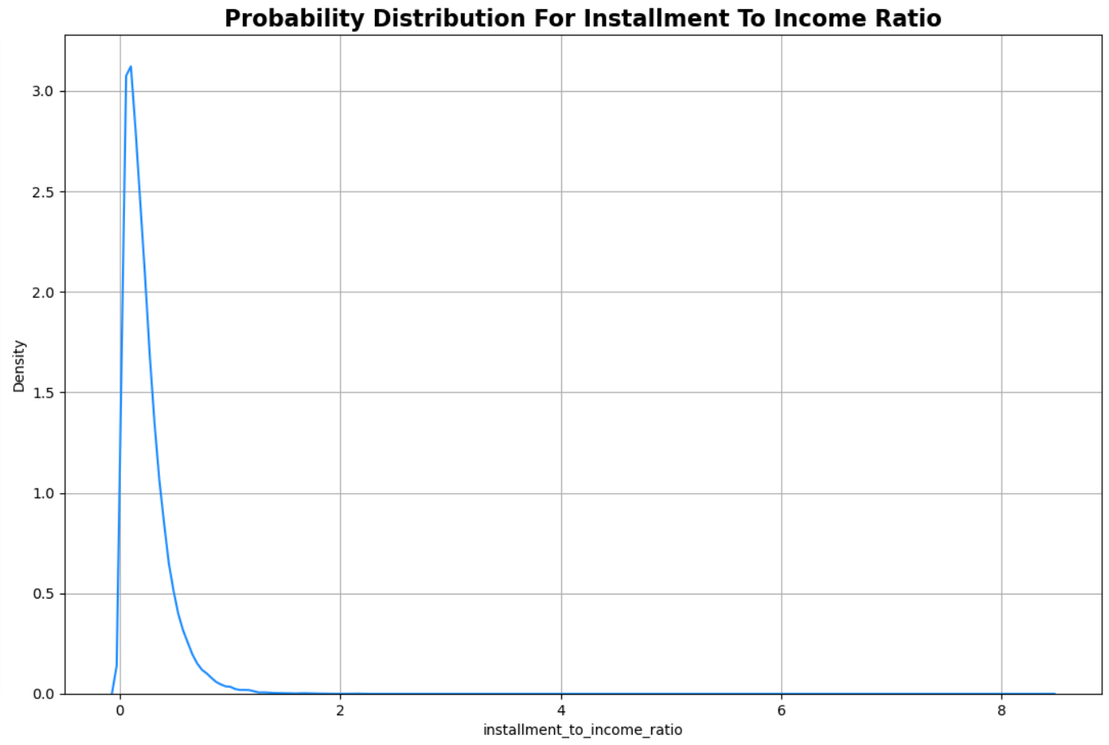
Out[43]:

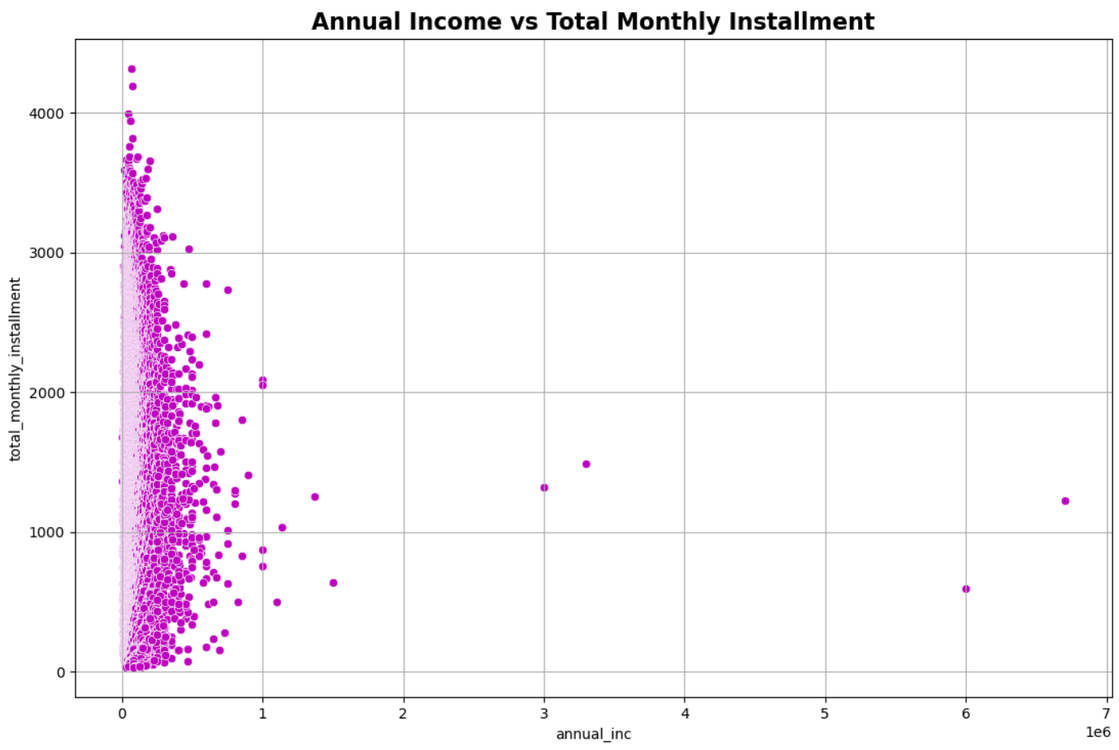
	borrower_id	total_monthly_installment	annual_inc	installment_to_income_ratio
0	16010	1681.88	2400.0	8.41
1	37255	2903.22	7100.0	4.91
2	40431	1361.37	5000.0	3.27
3	23240	1428.14	7111.0	2.41
4	42145	2720.97	15000.0	2.18
5	44725	2151.29	12000.0	2.15
6	32408	3589.95	20000.0	2.15
7	35725	2468.14	14000.0	2.12
8	17725	3122.30	18000.0	2.08
9	13577	3050.43	19000.0	1.93
10	30723	1722.83	11000.0	1.88
11	44138	1931.21	12500.0	1.85
12	26998	2150.41	14000.0	1.84
13	35127	2269.99	15000.0	1.82
14	17623	2406.51	16000.0	1.80

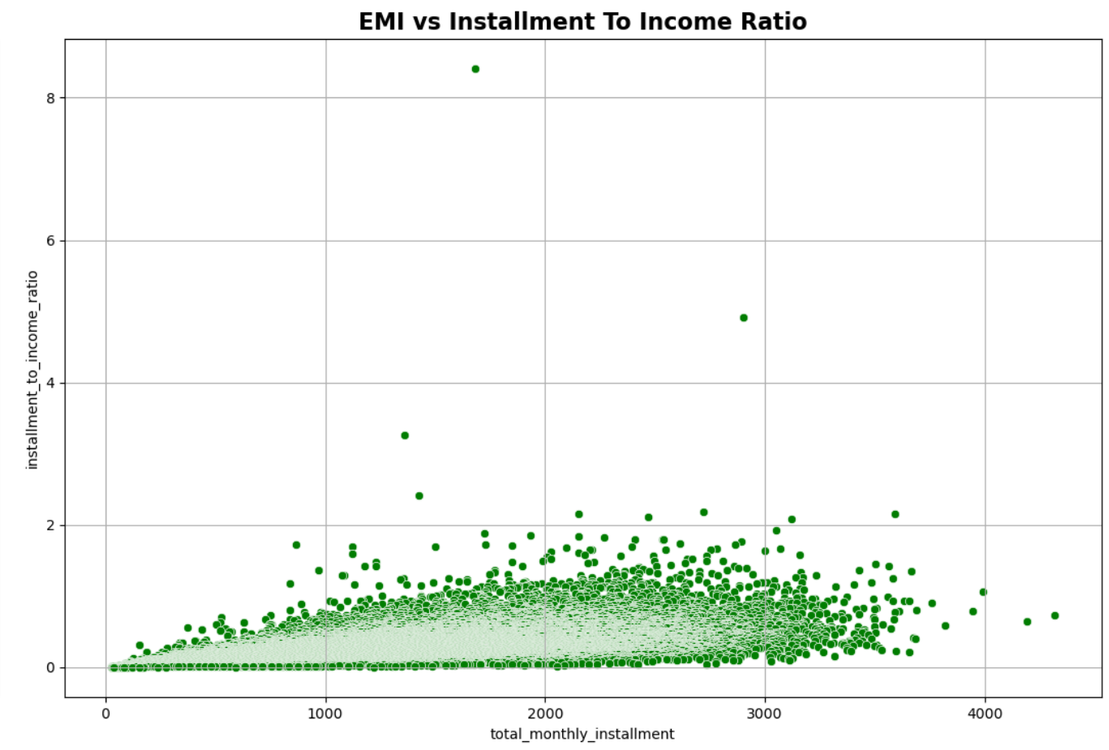
In [44]:

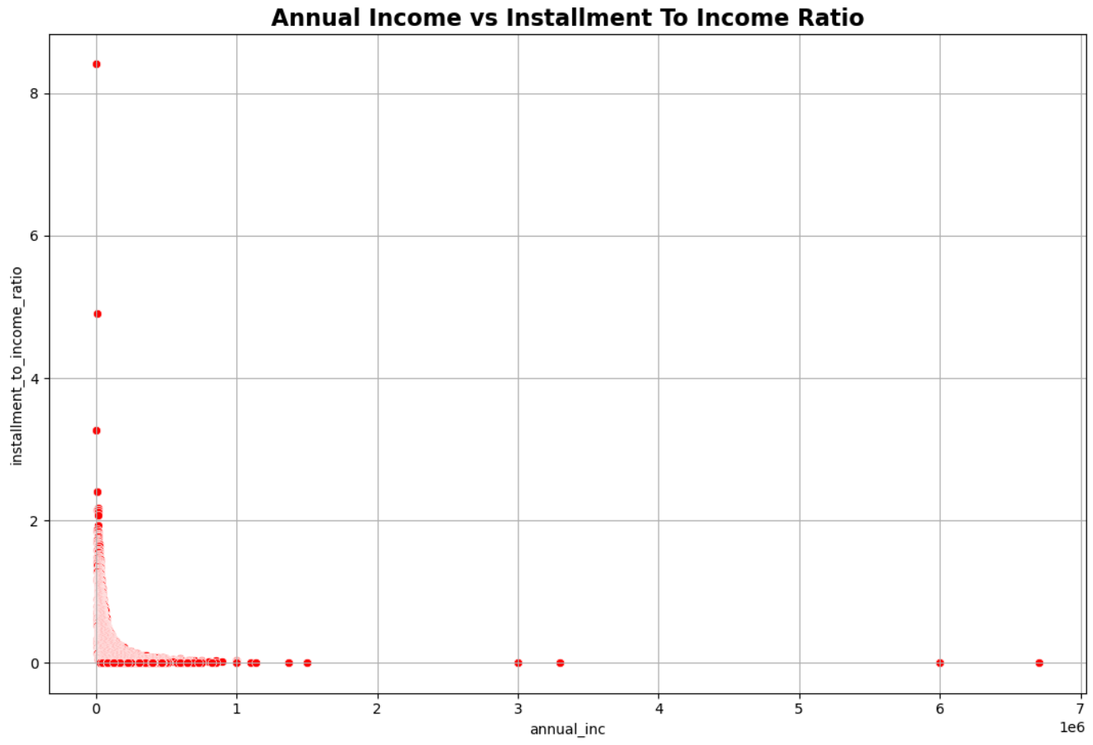
```
fig,ax = plt.subplots(nrows = 3,ncols = 2,figsize = (22,22))

sns.kdeplot(data = df,x = 'total_monthly_installment',color = 'crimson',ax = ax[0,0])
ax[0,0].grid(True)
ax[0,0].set_title('Probability Distribution For Total Monthly Installment',fontweight = 'bold',fontsize = '16')

sns.kdeplot(data = df,x = 'installment_to_income_ratio',color = 'dodgerblue',ax = ax[0,1])
ax[0,1].grid(True)
ax[0,1].set_title('Probability Distribution For Installment To Income Ratio',fontweight = 'bold',fontsize = '16')

sns.scatterplot(data = df,x = 'annual_inc',y = 'total_monthly_installment',ax = ax[1,0],color = 'm')
ax[1,0].set_title('Annual Income vs Total Monthly Installment',fontweight = 'bold',fontsize = '16')
ax[1,0].grid(True)

sns.scatterplot(data = df,x = 'total_monthly_installment',y = 'installment_to_income_ratio',ax = ax[1,1],color = 'g')
ax[1,1].set_title('EMI vs Installment To Income Ratio',fontweight = 'bold',fontsize = '16')
ax[1,1].grid(True)

sns.scatterplot(data = df,x = 'annual_inc',y = 'installment_to_income_ratio',ax = ax[2,0],color = 'r')
ax[2,0].set_title('Annual Income vs Installment To Income Ratio',fontweight = 'bold',fontsize = '16')
ax[2,0].grid(True)

corr = df[['total_monthly_installment','annual_inc','installment_to_income_ratio']].corr()
sns.heatmap(corr,annot = True,cmap = 'viridis',cbar = True,
            ax = ax[2,1],linewidths = 0.5)
ax[2,1].set_title('Correlation Heatmap',fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```

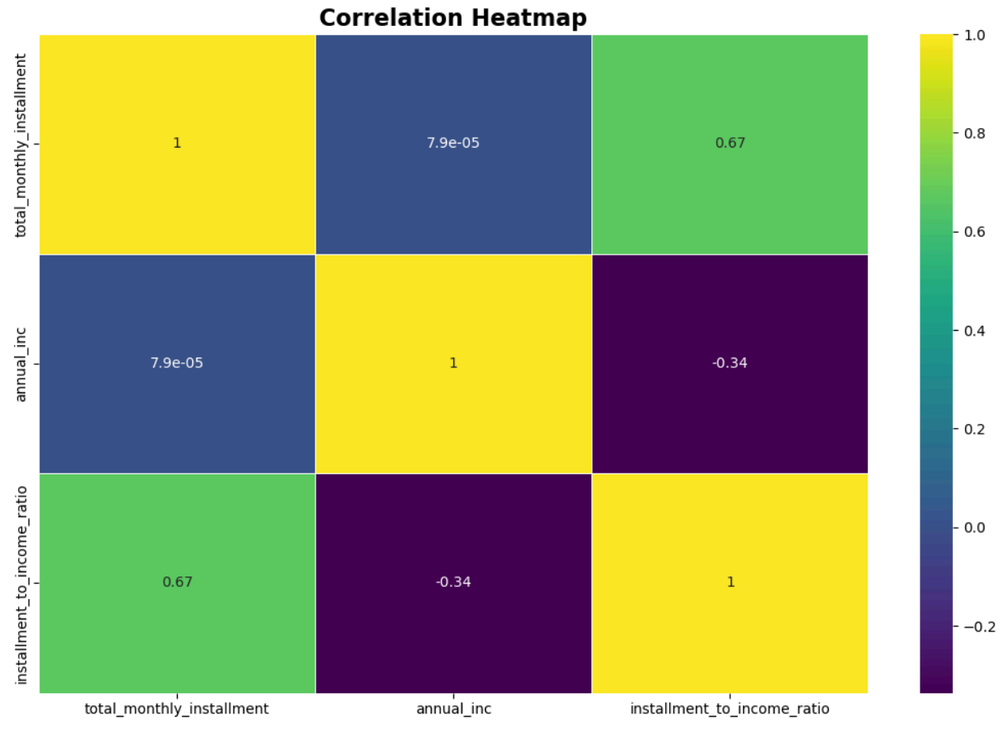












16. High Revolving Utilization(>90) & Costly Loans. Identifies borrowers maxing out their credit lines with costly loans and defaulting.

The analysis of high revolving utilization borrowers highlights a critical risk segment. From the dataframe, we see borrowers maxing out their credit lines (>90% revol_util) often coupled with high interest rates, with many loans marked as charged off or late. The histogram shows that interest rates for these loans are skewed toward the higher end, emphasizing their costly nature. The scatter plot illustrates a clear trend: higher revolving utilization generally coincides with higher interest rates, and longer-term loans add further risk. The bar plot of the top 15 ranked costly loans reinforces this, showing individual borrowers facing extreme interest rates and repayment stress. Overall, this segment represents the most financially vulnerable borrowers, needing close monitoring.

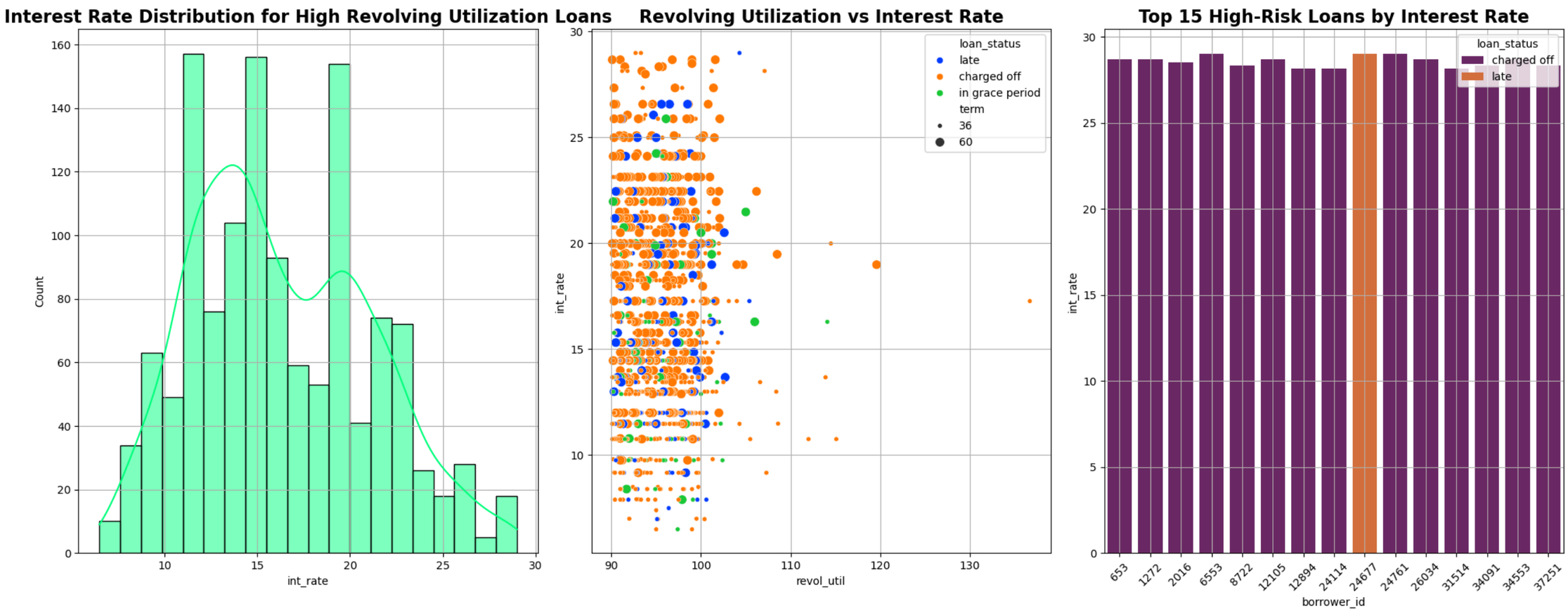
In [45]:
```
query = """SELECT *, DENSE_RANK() OVER(ORDER BY int_rate DESC,revol_util DESC) AS 'costly_loan_rank' FROM loans
           WHERE revol_util > 90 AND loan_status IN ('late','charged off','in grace period');"""

df = run_query(conn,query)
df
```

Out[45]:

	loan_id	loan_amnt	term	int_rate	grade	purpose	dti	delinq_2yrs	revol_util	loan_status	borrower_id	costly_loan_rank
0	LN061658	10475	36	28.99	G	home_improvement	21.09	0	104.30	late	24677	1
1	LN016270	4775	36	28.99	G	debt_consolidation	32.89	0	93.30	charged off	6553	2
2	LN061865	35000	36	28.99	G	debt_consolidation	24.66	0	92.70	charged off	24761	3
3	LN001684	12275	60	28.67	G	debt_consolidation	37.32	0	101.60	charged off	653	4
4	LN003173	16150	60	28.67	G	other	44.56	0	99.00	charged off	1272	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1285	LN051127	8000	36	6.97	A	credit_card	33.42	0	95.20	charged off	20476	1089
1286	LN088179	10000	36	6.97	A	credit_card	15.58	0	95.10	late	35254	1090
1287	LN092006	22000	36	6.49	A	debt_consolidation	14.20	0	99.00	charged off	36792	1091
1288	LN084534	12000	36	6.49	A	credit_card	25.56	0	97.40	in grace period	33814	1092
1289	LN087297	28000	36	6.49	A	debt_consolidation	19.90	1	95.00	charged off	34893	1093

1290 rows × 12 columns

In [46]:
```
fig,ax  = plt.subplots(nrows = 1,ncols = 3,figsize = (20,8))

sns.histplot(df['int_rate'], bins = 20, kde = True,ax = ax[0],color = 'springgreen')
ax[0].set_title("Interest Rate Distribution for High Revolving Utilization Loans",fontweight = 'bold',fontsize = '16')
ax[0].grid(True)

sns.scatterplot(data = df,x = 'revol_util',y = 'int_rate',hue = 'loan_status',size  = 'term',palette = 'bright',ax = ax[1])
ax[1].grid(True)
ax[1].set_title('Revolving Utilization vs Interest Rate',fontweight = 'bold',fontsize = '16')

# Bar plot of the top 15 ranked loans by interest rate
top_15 = df[df['costly_loan_rank'] <= 15]
sns.barplot(data = top_15,x = 'borrower_id',y = 'int_rate',hue = 'loan_status',palette = 'inferno',ax = ax[2])
ax[2].grid(True)
ax[2].tick_params(axis='x', rotation = 45)
ax[2].set_title('Top 15 High-Risk Loans by Interest Rate',fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```



17. Purpose + Term Loan Rates. Explores how loan costs vary jointly by purpose and repayment duration.

The analysis of loan costs by purpose and term reveals clear patterns in both interest rates and loan amounts. From the dataframe, longer-term loans (60 months) consistently carry higher interest rates and larger average amounts compared to 36-month loans across nearly all purposes. For example, car and credit card loans almost double in size and increase in rate with longer terms, while niche purposes like renewable energy or small business also show elevated costs for extended terms. The bar plots highlight these trends visually, showing higher interest and loan amounts for 60-month loans across all purposes, while the color-coded distinction makes it easy to compare between different purposes. This indicates that both purpose and term are key drivers of loan pricing and risk exposure in the portfolio.

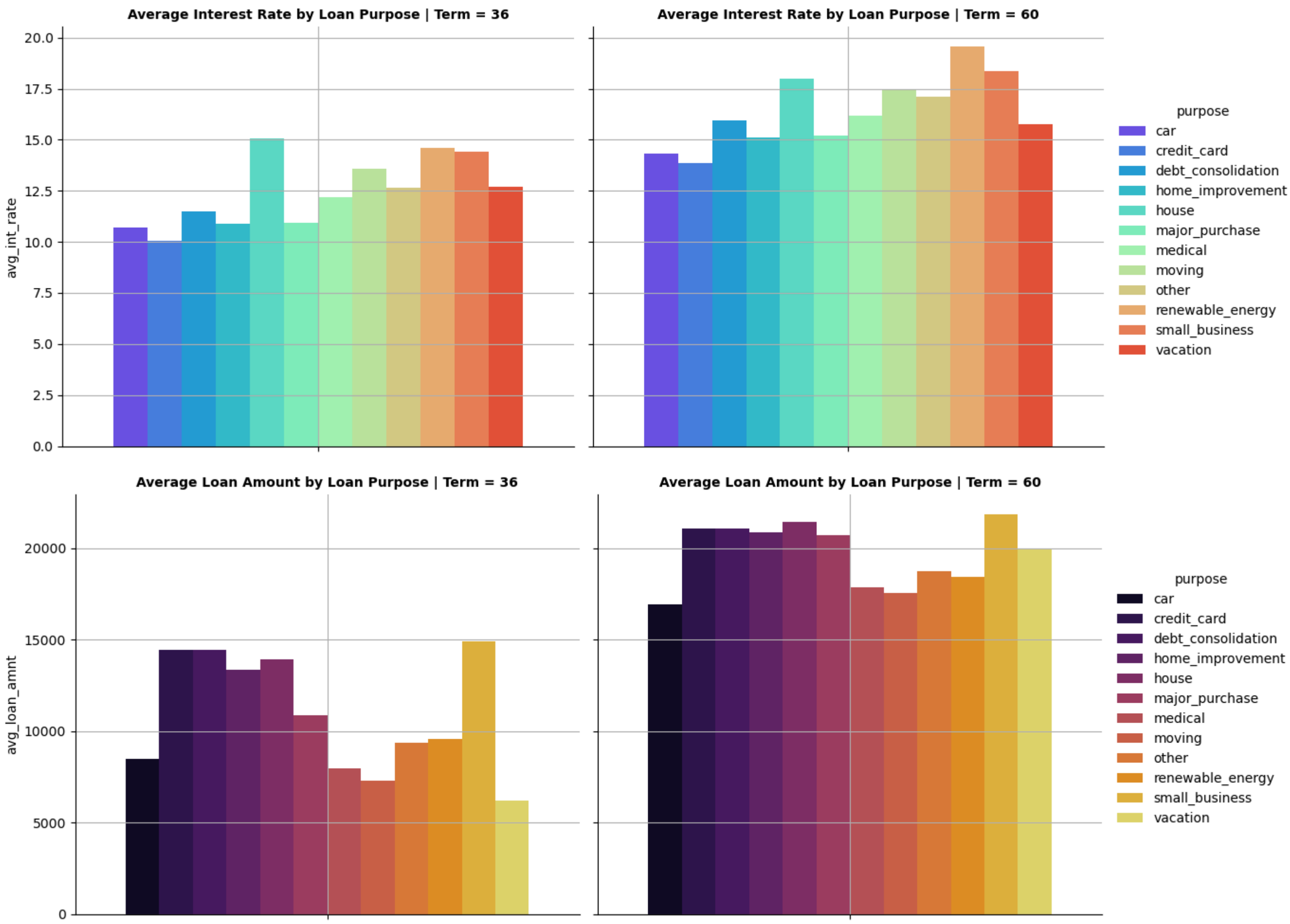
In [47]:
```
query = """SELECT purpose,term,ROUND(AVG(int_rate),2) AS 'avg_int_rate',ROUND(AVG(loan_amnt),2) AS 'avg_loan_amnt'
           FROM loans
           GROUP BY purpose,term
           ORDER BY purpose,term;"""

df = run_query(conn,query)
df
```

Out[47]:

	purpose	term	avg_int_rate	avg_loan_amnt
0	car	36	10.71	8469.46
1	car	60	14.32	16946.40
2	credit_card	36	10.03	14428.14
3	credit_card	60	13.88	21072.80
4	debt_consolidation	36	11.51	14457.53
5	debt_consolidation	60	15.96	21108.40
6	home_improvement	36	10.89	13350.88
7	home_improvement	60	15.09	20891.05
8	house	36	15.07	13921.70
9	house	60	18.00	21457.99
10	major_purchase	36	10.93	10850.56
11	major_purchase	60	15.21	20703.29
12	medical	36	12.19	7963.42
13	medical	60	16.17	17857.37
14	moving	36	13.56	7298.69
15	moving	60	17.49	17581.19
16	other	36	12.66	9384.75
17	other	60	17.10	18738.61
18	renewable_energy	36	14.59	9543.75
19	renewable_energy	60	19.55	18422.22
20	small_business	36	14.40	14933.33
21	small_business	60	18.34	21840.95
22	vacation	36	12.71	6212.04
23	vacation	60	15.77	19944.26

In [48]:
```
g1 = sns.catplot(data = df,hue = 'purpose',y = 'avg_int_rate',kind = 'bar',palette = 'rainbow',col = 'term',height = 5,aspect = 1.2)
g1.set_titles("Average Interest Rate by Loan Purpose | Term = {col_name}", fontweight = 'bold',fontsize = '16')
for ax in g1.axes.flat:
    ax.grid(True)

g2 = sns.catplot(data = df,hue = 'purpose',y = 'avg_loan_amnt',kind = 'bar',palette = 'inferno',col = 'term',height = 5,aspect = 1.2)
g2.set_titles("Average Loan Amount by Loan Purpose | Term = {col_name}", fontweight = 'bold',fontsize = '16')
for ax in g2.axes.flat:
```

```
    ax.grid(True)

plt.show()
```



18. Grade-Purpose Portfolio Share. Reveals diversification or concentration in loan offerings.

The grade-purpose portfolio analysis shows how loans are distributed across grades and purposes, highlighting both diversification and concentration. Most loans are concentrated in debt consolidation and credit card purposes, especially in grades A, B, and C, while niche purposes like small business, home improvement, and major purchase appear mostly in lower grades (D–G) with smaller portfolio shares. Average loan amounts tend to increase slightly for lower grades and specialized purposes, reflecting higher risk and cost. The stacked bar plots make these patterns clear: one plot shows average loan amounts by grade and purpose, emphasizing larger loans in riskier grades, while the second plot shows the percentage of total loans per grade-purpose combination, highlighting that a few combinations dominate the portfolio. Overall, the portfolio is moderately diversified but heavily weighted toward mainstream consumer lending.

In [49]:
```
query = """SELECT t2.grade,t2.purpose,t2.avg_loan_amnt,t2.percent_loans_per_grade_purpose_combo
           FROM (SELECT * , DENSE_RANK() OVER(PARTITION BY t1.grade ORDER BY t1.avg_loan_amnt DESC) AS 'purpose_rank'
                 FROM (SELECT grade,purpose,ROUND(AVG(loan_amnt),2) AS 'avg_loan_amnt',
                        ROUND(COUNT(loan_id) * 100 / (SELECT COUNT(loan_id) FROM loans),2) AS 'percent_loans_per_grade_purpose_combo'
                         FROM loans
                         GROUP BY grade,purpose
                         ORDER BY grade,purpose) t1) t2
           WHERE t2.purpose_rank <= 3;"""

df = run_query(conn,query)
df
```

Out[49]:

	grade	purpose	avg_loan_amnt	percent_loans_per_grade_purpose_combo
0	A	house	18836.22	0.04
1	A	debt_consolidation	16050.57	9.75
2	A	credit_card	15857.61	6.54
3	B	small_business	15459.51	0.21
4	B	credit_card	15209.26	8.52
5	B	debt_consolidation	15169.49	16.56
6	C	credit_card	16901.46	5.74
7	C	debt_consolidation	16565.83	16.59
8	C	small_business	15758.00	0.30
9	D	home_improvement	17964.22	0.77
10	D	debt_consolidation	17744.42	7.64
11	D	credit_card	17688.89	1.96
12	E	home_improvement	19833.13	0.40
13	E	debt_consolidation	19161.10	4.72
14	E	credit_card	19146.97	0.94
15	F	home_improvement	21587.56	0.16
16	F	major_purchase	20922.22	0.05
17	F	debt_consolidation	20217.61	1.78
18	G	small_business	21033.62	0.02
19	G	home_improvement	19790.10	0.04
20	G	major_purchase	19304.69	0.01

In [50]:
```
pivot_df1 = df.pivot(index = 'grade',
                     columns = 'purpose',
                     values = 'avg_loan_amnt').fillna(0)
pivot_df1 = pivot_df1.astype('float')
pivot_df1
```

Out[50]:

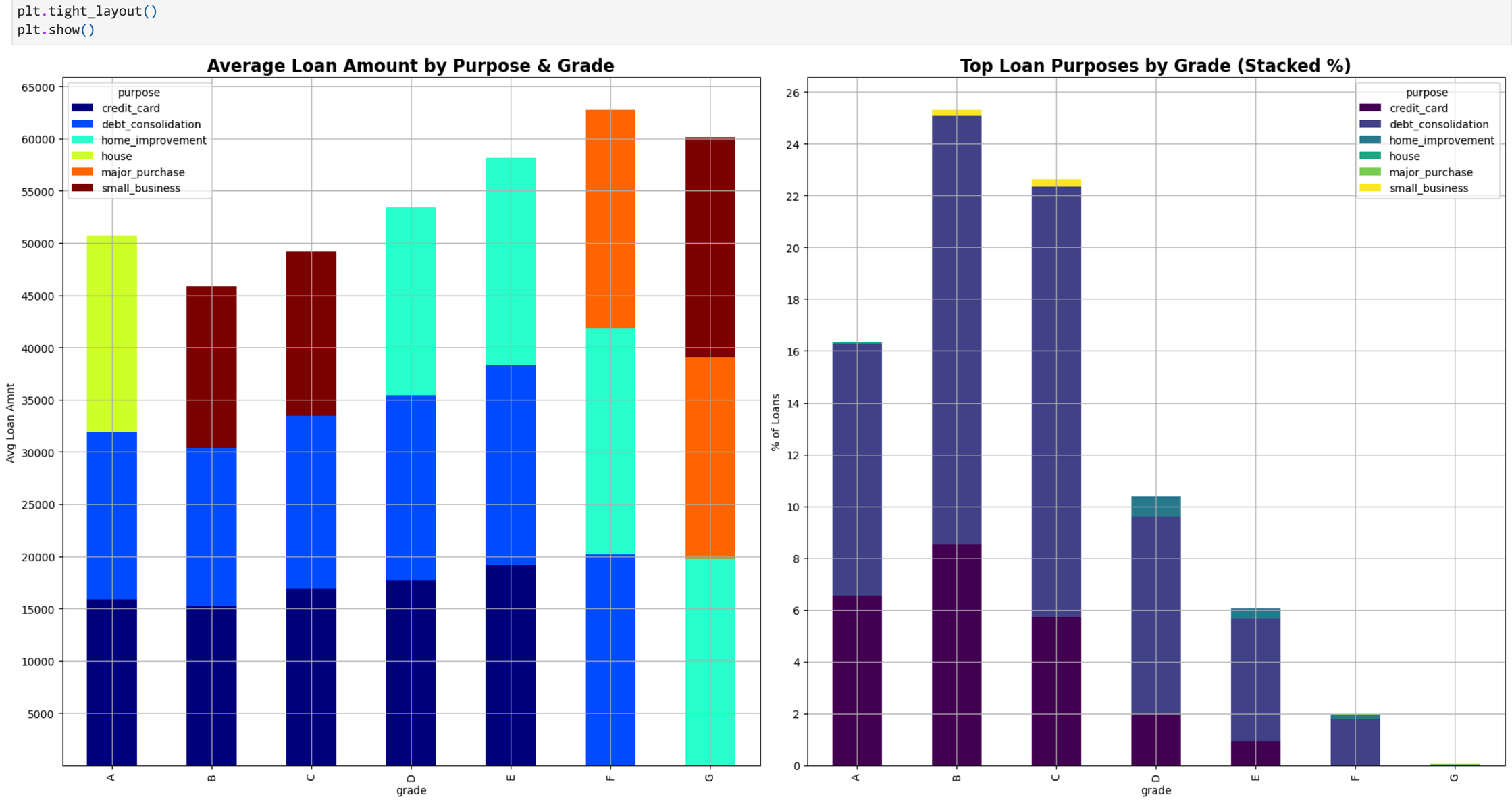
grade	credit_card	debt_consolidation	home_improvement	house	major_purchase	small_business
A	15857.61	16050.57	0.00	18836.22	0.00	0.00
B	15209.26	15169.49	0.00	0.00	0.00	15459.51
C	16901.46	16565.83	0.00	0.00	0.00	15758.00
D	17688.89	17744.42	17964.22	0.00	0.00	0.00
E	19146.97	19161.10	19833.13	0.00	0.00	0.00
F	0.00	20217.61	21587.56	0.00	20922.22	0.00
G	0.00	0.00	19790.10	0.00	19304.69	21033.62

In [51]:
```
pivot_df2 = df.pivot(index = 'grade',
                     columns = 'purpose',
                     values = 'percent_loans_per_grade_purpose_combo').fillna(0)
pivot_df2 = pivot_df2.astype('float')
pivot_df2
```

Out[51]:

grade	credit_card	debt_consolidation	home_improvement	house	major_purchase	small_business
A	6.54	9.75	0.00	0.04	0.00	0.00
B	8.52	16.56	0.00	0.00	0.00	0.21
C	5.74	16.59	0.00	0.00	0.00	0.30
D	1.96	7.64	0.77	0.00	0.00	0.00
E	0.94	4.72	0.40	0.00	0.00	0.00
F	0.00	1.78	0.16	0.00	0.05	0.00
G	0.00	0.00	0.04	0.00	0.01	0.02

In [52]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (20,10))

ax[0] = pivot_df1.plot(kind = "bar", stacked = True, colormap = 'jet',ax = ax[0])
ax[0].grid(True)
ax[0].set_ylabel("Avg Loan Amnt")
ax[0].set_yticks(ticks = range(5000,70000,5000))
ax[0].set_title("Average Loan Amount by Purpose & Grade",fontweight = 'bold',fontsize = '16')

ax[1] = pivot_df2.plot(kind = "bar", stacked = True, colormap = 'viridis',ax = ax[1])
ax[1].grid(True)
ax[1].set_ylabel("% of Loans")
ax[1].set_yticks(ticks = range(0,28,2))
ax[1].set_title("Top Loan Purposes by Grade (Stacked %)",fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```



In [ ]:

In [ ]:

C. Loan Performance & Risk Analysis- How well are the loans performing, and where are the risks?

Not every loan performs the same – some are repaid, some are delayed, and some default. This section explores repayment outcomes and identifies high-risk loan segments.

Summary-

The loan portfolio overall is performing well, with most loans being current or fully paid; however, risk is concentrated in specific segments. Short-term loans (36-month) in mid-grades B, C, and D dominate defaults, while longer-term loans (60-month) contribute to risk in higher grades like E and F. Certain loan purposes—small business, medical,

and debt consolidation—show higher default percentages, indicating that both borrower profile and loan purpose jointly influence risk. Past delinquencies emerge as a strong predictor of default: borrowers with 3–4 prior delinquencies account for a major portion of defaults, particularly when combined with high revolving utilization and moderate-to-high debt-to-income ratios, reflecting compounded financial stress. Interestingly, income bands have little effect on default rates, although wealthier borrowers tend to hold more accounts, which may subtly increase exposure. Higher interest rates correlate with higher defaults, especially among weaker credit grades and longer-term loans, demonstrating that repayment risk rises with borrowing cost. Grade migration analysis shows that while high-grade borrowers often maintain or improve their status, mid- and low-grade borrowers frequently experience deteriorating creditworthiness, signaling the need for ongoing monitoring. Finally, lenders adjust interest rates based on perceived borrower risk: most see stable or slightly increased rates over successive loans, reinforcing the link between repayment behavior and pricing strategy. Mid-grade, short-term loans for high-risk purposes, compounded by prior delinquencies and high financial leverage, form the core of portfolio risk, while income alone is not a strong predictor—highlighting the importance of multi-factor risk assessment, credit monitoring, and adaptive pricing strategies.

In [ ]:

19. Loan Status Breakdown. Key metric showing defaults, charge-offs, and active loans, also reveals if certain grades or terms dominate particular loan statuses.

In the Loan Status Breakdown, most loans are performing well, with the majority being either current or fully paid, while charge-offs remain relatively low. Short-term (36-month) loans generally dominate in volume, especially in mid-grades like B and C. The highest default rates are seen in grade C (36-month term, 2.44%), followed by B (36-month, 1.85%) and D (36-month, 1.43%). Longer-term loans (60 months) have slightly lower percentages of charge-offs but still contribute to risk in higher grades like E and F. The bar plots clearly show the distribution of total loans and percentages across grades and terms, highlighting where defaults concentrate, while the heatmap effectively visualizes charged-off percentages, making it easy to identify the riskiest grade-term combinations. The top 15 segments bar plot confirms that 36-month loans in grades B, C, and D are the main contributors to credit risk.

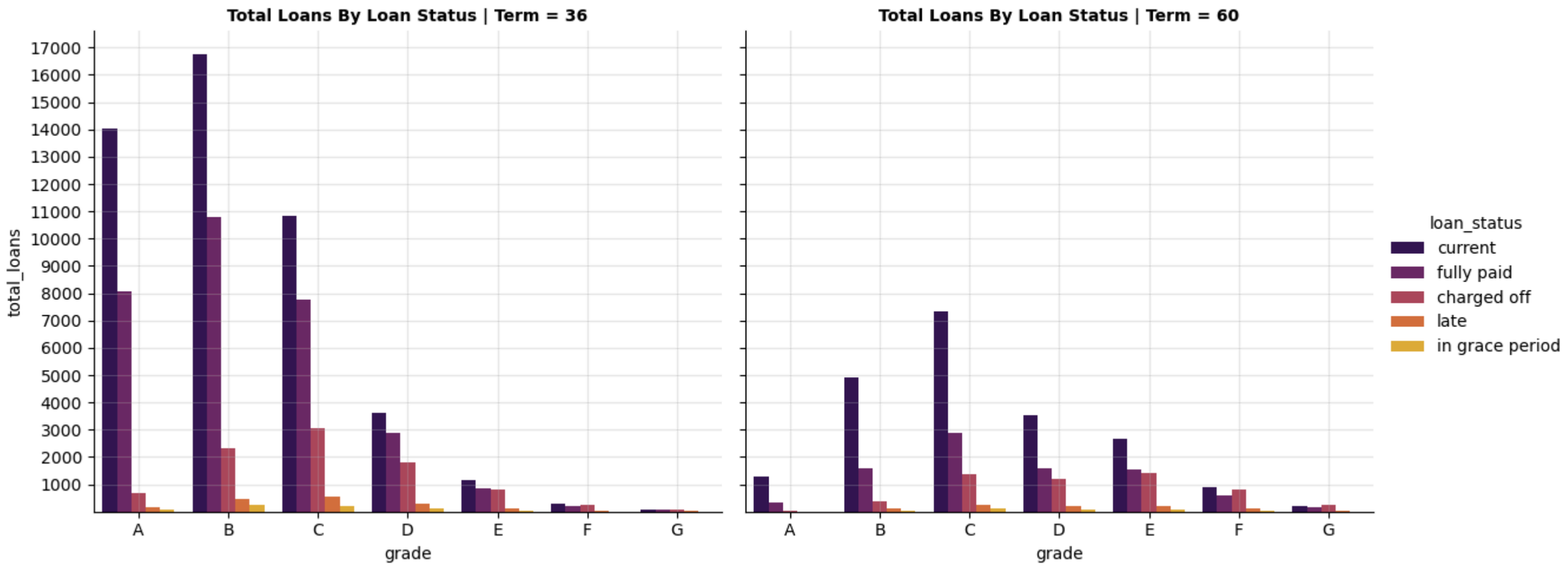
In [53]:
```
query = """SELECT grade,term,loan_status,COUNT(loan_id) AS 'total_loans',
           ROUND(COUNT(loan_id) * 100/ (SELECT COUNT(loan_id) FROM loans),2) As 'percent_loans'
           FROM loans
           GROUP BY grade,term,loan_status
           ORDER BY grade,term,total_loans DESC;"""

df = run_query(conn,query)
df
```

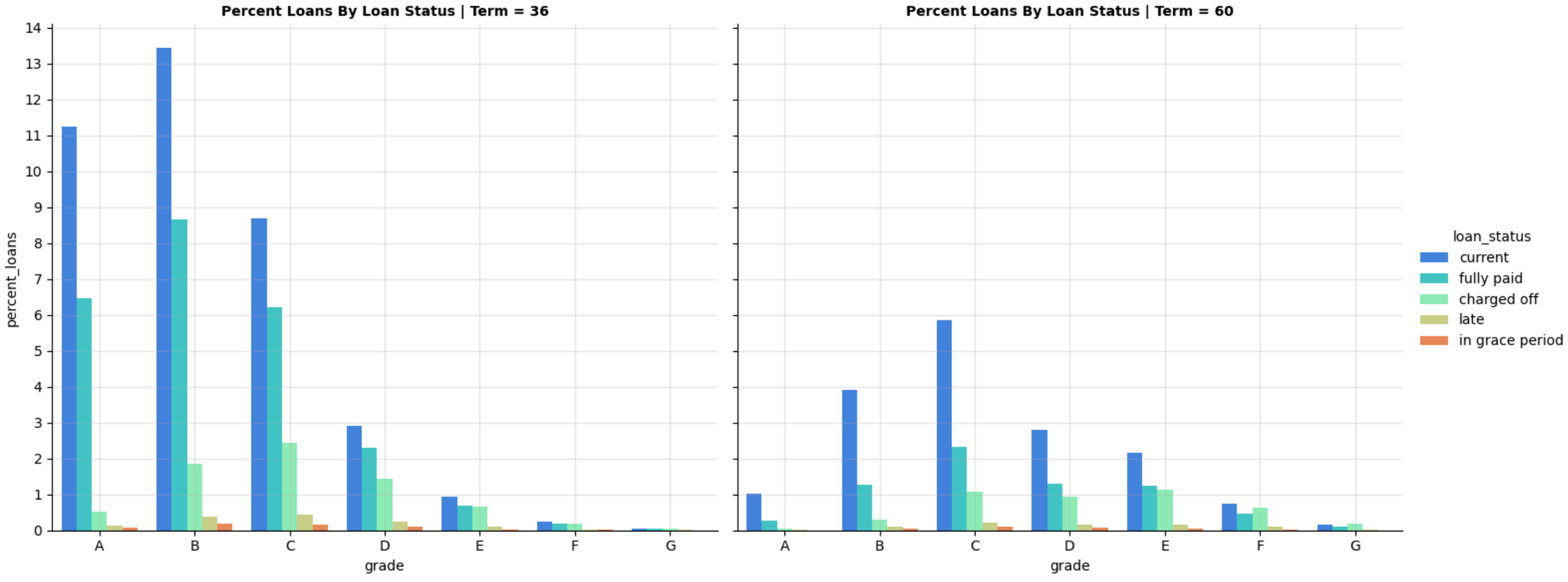
Out[53]:

	grade	term	loan_status	total_loans	percent_loans
0	A	36	current	14033	11.24
1	A	36	fully paid	8076	6.47
2	A	36	charged off	661	0.53
3	A	36	late	179	0.14
4	A	36	in grace period	84	0.07
...	...	...	...	...	...
65	G	60	charged off	240	0.19
66	G	60	current	188	0.15
67	G	60	fully paid	141	0.11
68	G	60	late	25	0.02
69	G	60	in grace period	6	0.00

70 rows × 5 columns

In [54]:
```
g1 = sns.catplot(data = df,hue = 'loan_status',x = 'grade',y = 'total_loans',kind = 'bar',palette = 'inferno',
                 col = 'term',height = 5,aspect = 1.2)
g1.set_titles("Total Loans By Loan Status | Term = {col_name}", fontweight = 'bold',fontsize = '18')
for ax in g1.axes.flat:
    ax.grid(True, alpha=0.3)
    ax.set_yticks(ticks = range(1000,18000,1000))


g2 = sns.catplot(data = df,hue = 'loan_status',x = 'grade',y = 'percent_loans',kind = 'bar',palette = 'rainbow',col = 'term',
                 height = 6,aspect = 1.2)
g2.set_titles("Percent Loans By Loan Status | Term = {col_name}", fontweight = 'bold',fontsize = '18')
for ax in g2.axes.flat:
    ax.grid(True, alpha=0.3)
    ax.set_yticks(ticks = range(0,15))

plt.show()
```





In [55]:
```
heatmap_df = df[df["loan_status"] == "charged off"].pivot(index = "grade", columns = "term", values = "percent_loans")
heatmap_df = heatmap_df.astype(float)
heatmap_df
```

Out[55]:

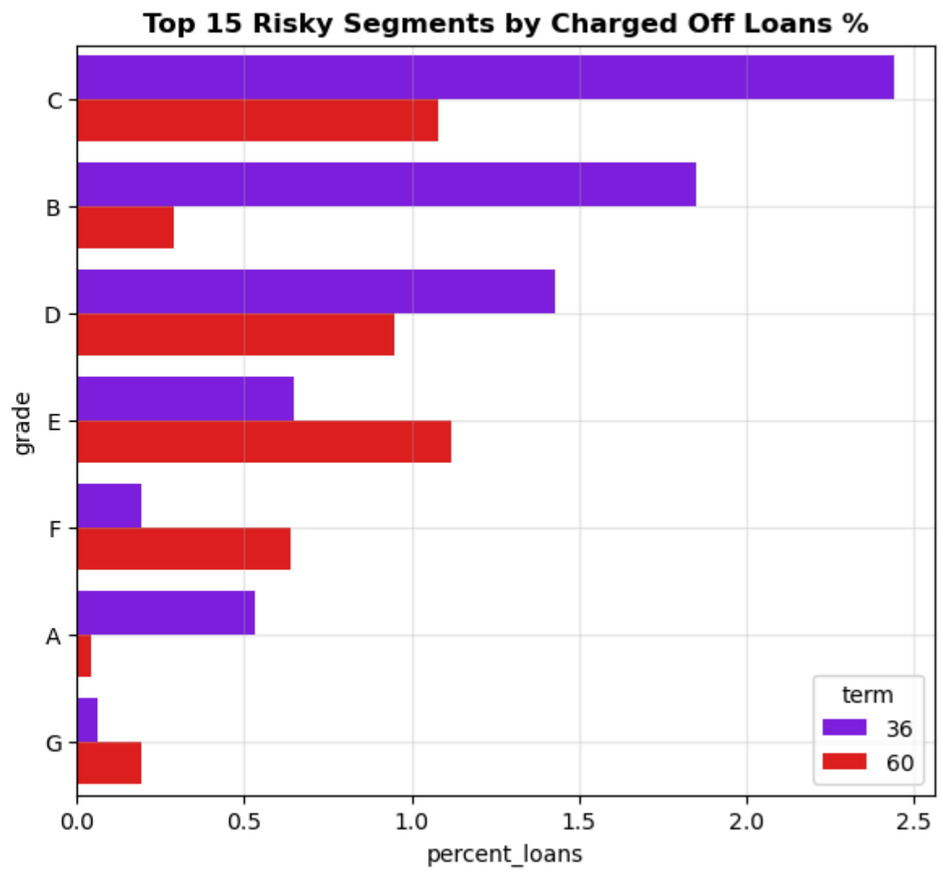
grade	36	60
A	0.53	0.04
B	1.85	0.29
C	2.44	1.08
D	1.43	0.95
E	0.65	1.12
F	0.19	0.64
G	0.06	0.19

In [56]:
```
df['percent_loans'] = df['percent_loans'].astype('float')
top_defaults = df[(df["loan_status"] == "charged off")].nlargest(15, "percent_loans")
top_defaults
```

Out[56]:

	grade	term	loan_status	total_loans	percent_loans
22	C	36	charged off	3043	2.44
12	B	36	charged off	2307	1.85
32	D	36	charged off	1784	1.43
47	E	60	charged off	1395	1.12
27	C	60	charged off	1350	1.08
37	D	60	charged off	1185	0.95
42	E	36	charged off	814	0.65
56	F	60	charged off	794	0.64
2	A	36	charged off	661	0.53
17	B	60	charged off	363	0.29
65	G	60	charged off	240	0.19
51	F	36	charged off	241	0.19
60	G	36	charged off	81	0.06
7	A	60	charged off	50	0.04

In [57]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))

sns.heatmap(heatmap_df, annot = True, fmt = ".2f",cmap = "inferno",ax = ax[0])
ax[0].set_title("Charged Off Loan % by Grade and Term", fontweight = "bold")

sns.barplot(data = top_defaults, x = "percent_loans", y = "grade", hue = "term", palette = "rainbow",ax = ax[1])
ax[1].set_title("Top 15 Risky Segments by Charged Off Loans %",fontweight = "bold")
ax[1].grid(True, alpha=0.3)
plt.show()
```



20. Default % by Purpose and Grade. Certain purposes (e.g., small business, medical) tend to default more; grading within purpose adds granularity for risk management.

In the Default % by Purpose and Grade analysis, we see that certain loan purposes consistently have higher defaults, with small business, medical, and debt consolidation showing elevated risk. Grade C and B loans contribute the most to total charge-offs across most purposes, indicating higher-risk borrowers are concentrated in these grades. While low-grade loans like F and G have fewer total defaults, their percentages remain meaningful for specific purposes. The stacked bar plots make it easy to compare both total charged-off loans and their proportion within each grade, highlighting that mid-grade loans for debt consolidation and credit card purposes form the largest default segments, while purposes like renewable energy and vacation have minimal impact. Overall, this analysis helps identify which grade-purpose combinations require closer risk monitoring.

In [58]:
```
query = """SELECT purpose,grade,COUNT(loan_id) AS 'total_charged_off_loans',
           ROUND(COUNT(loan_id) * 100 / (SELECT COUNT(loan_id) FROM loans WHERE loan_status = 'charged off'),2) AS 'percent_charged_off_loans'
           FROM loans
           WHERE loan_status = 'charged off'
           GROUP BY purpose,grade
           ORDER BY total_charged_off_loans DESC;"""

df = run_query(conn,query)

pivot_df1 = df.pivot(index = 'grade',
                     columns = 'purpose',
                     values = 'total_charged_off_loans').fillna(0)
pivot_df1 = pivot_df1.astype('float')

pivot_df2 = df.pivot(index = 'grade',
                     columns = 'purpose',
                     values = 'percent_charged_off_loans').fillna(0)
pivot_df2 = pivot_df2.astype('float')
```

In [59]:
```
pivot_df1
```

Out[59]:

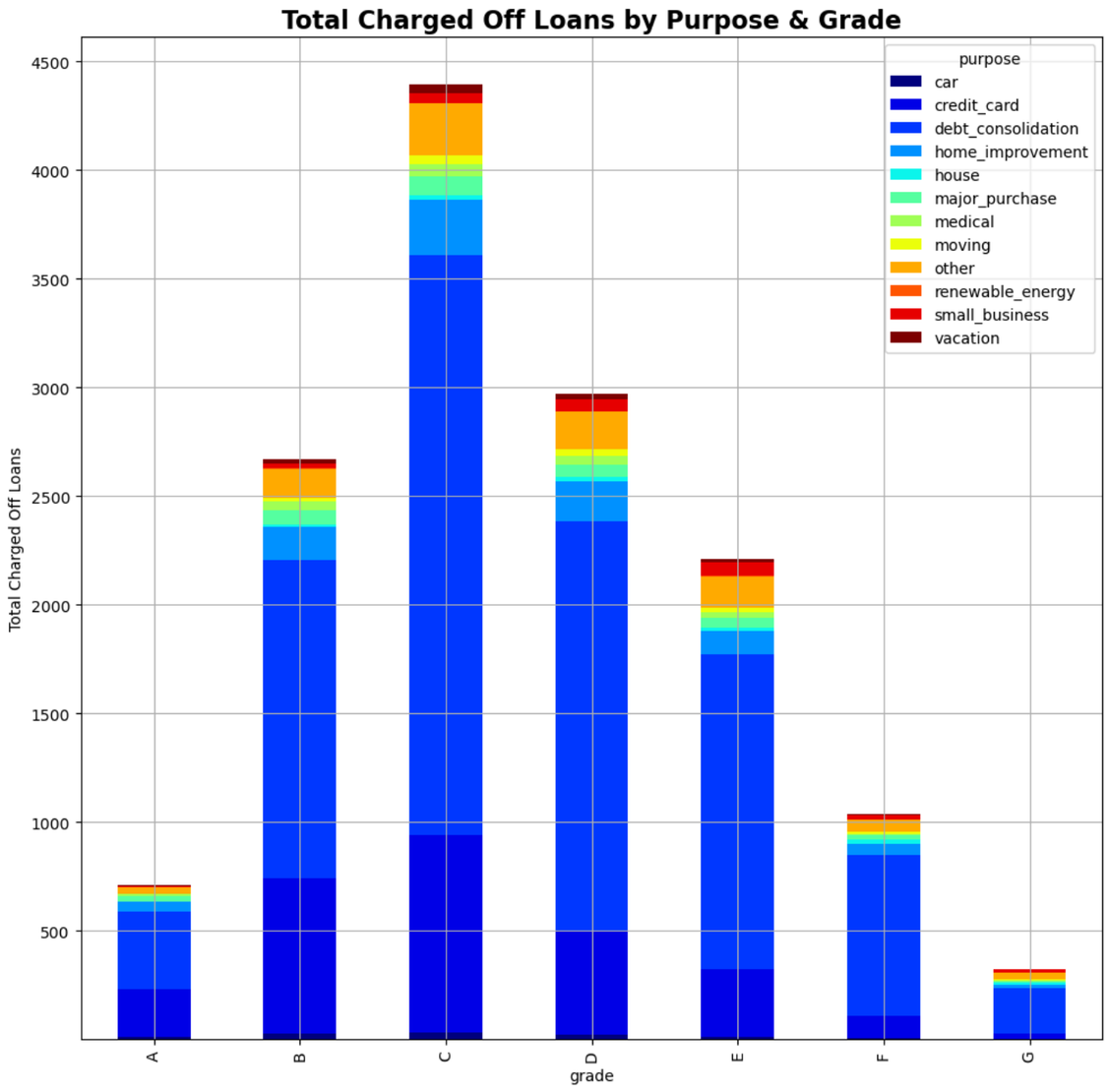
grade	car	credit_card	debt_consolidation	home_improvement	house	major_purchase	medical	moving	other	renewable_energy	small_business	vacation
A	11.0	220.0	355.0	46.0	0.0	24.0	10.0	5.0	27.0	1.0	3.0	9.0
B	23.0	716.0	1466.0	155.0	9.0	67.0	40.0	15.0	135.0	1.0	23.0	20.0
C	29.0	912.0	2669.0	255.0	18.0	89.0	57.0	38.0	239.0	3.0	45.0	39.0
D	21.0	474.0	1891.0	181.0	22.0	57.0	39.0	30.0	173.0	3.0	53.0	25.0
E	8.0	314.0	1451.0	106.0	16.0	46.0	25.0	22.0	142.0	3.0	63.0	13.0
F	4.0	101.0	741.0	53.0	19.0	21.0	4.0	12.0	54.0	2.0	22.0	2.0
G	2.0	24.0	210.0	15.0	10.0	5.0	3.0	5.0	34.0	0.0	13.0	0.0

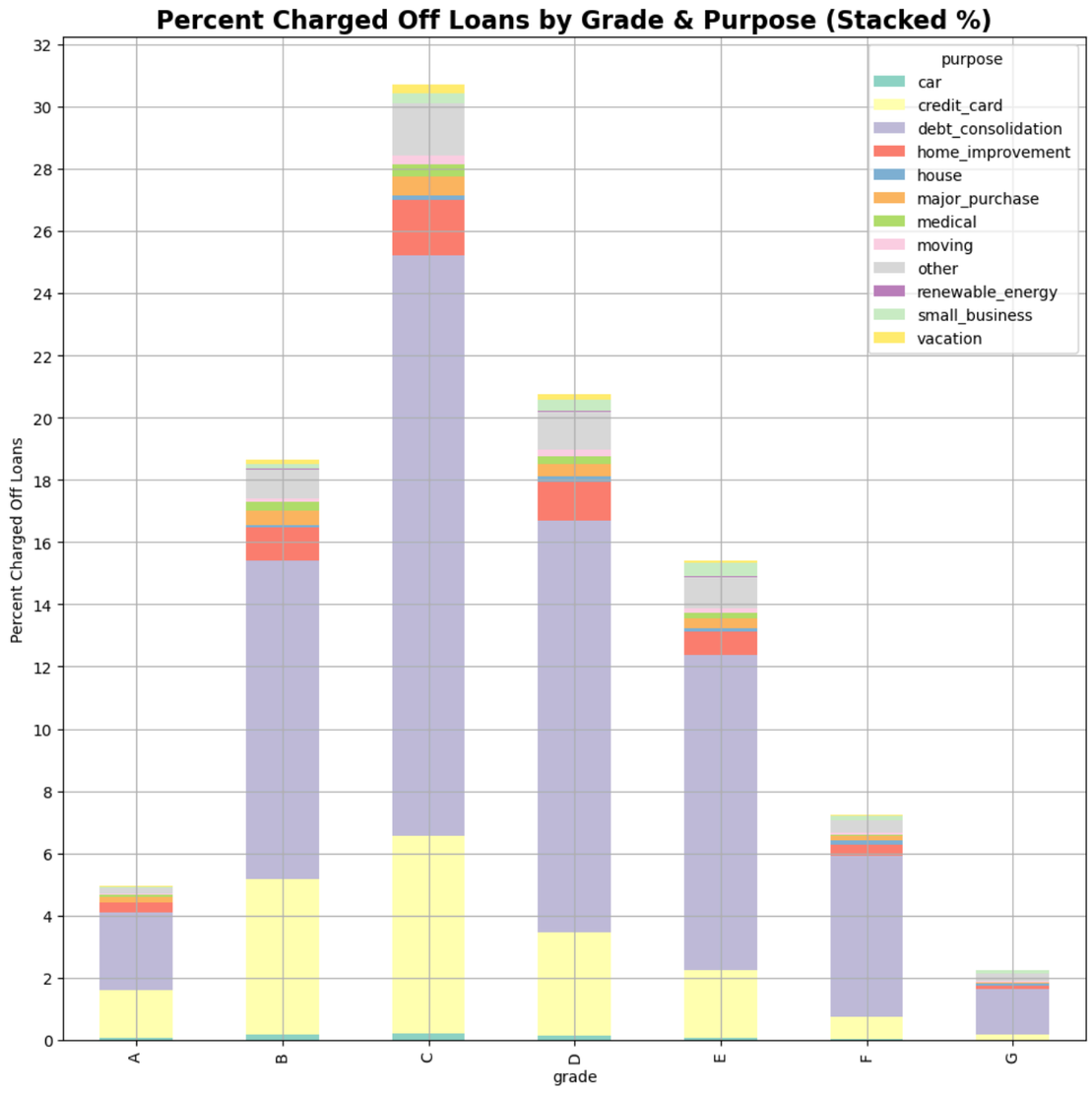
In [60]:
```
pivot_df2
```

Out[60]:

grade	car	credit_card	debt_consolidation	home_improvement	house	major_purchase	medical	moving	other	renewable_energy	small_business	vacation
A	0.08	1.54	2.48	0.32	0.00	0.17	0.07	0.03	0.19	0.01	0.02	0.06
B	0.16	5.00	10.25	1.08	0.06	0.47	0.28	0.10	0.94	0.01	0.16	0.14
C	0.20	6.37	18.65	1.78	0.13	0.62	0.40	0.27	1.67	0.02	0.31	0.27
D	0.15	3.31	13.22	1.27	0.15	0.40	0.27	0.21	1.21	0.02	0.37	0.17
E	0.06	2.19	10.14	0.74	0.11	0.32	0.17	0.15	0.99	0.02	0.44	0.09
F	0.03	0.71	5.18	0.37	0.13	0.15	0.03	0.08	0.38	0.01	0.15	0.01
G	0.01	0.17	1.47	0.10	0.07	0.03	0.02	0.03	0.24	0.00	0.09	0.00

In [61]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (20,10))

ax[0] = pivot_df1.plot(kind = "bar", stacked = True, colormap = 'jet',ax = ax[0])
ax[0].grid(True)
ax[0].set_ylabel("Total Charged Off Loans")
ax[0].set_yticks(ticks = range(500,5000,500))
ax[0].set_title("Total Charged Off Loans by Purpose & Grade",fontweight = 'bold',fontsize = '16')

ax[1] = pivot_df2.plot(kind = "bar", stacked = True, colormap = 'Set3',ax = ax[1])
ax[1].grid(True)
ax[1].set_ylabel("Percent Charged Off Loans")
ax[1].set_yticks(ticks = range(0,34,2))
ax[1].set_title("Percent Charged Off Loans by Grade & Purpose (Stacked %)",fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```





21. Past Delinquencies vs Default. Past delinquency is a strong predictor of future repayment failure; adding revol_util and dti to see compounded risk shows deeper risk signals.

The Past Delinquencies vs Default analysis shows that borrowers with prior delinquencies are significantly more likely to default, especially those with 3–4 delinquencies in the past two years, accounting for over 70% of charged-off loans in the top segments. Average revolving utilization hovers around 47–56% for most defaulted loans, while DTI remains moderately high (~18–21%), indicating that these borrowers are both heavily leveraged and carry substantial debt. The total number of loans defaults decreases sharply as delinquencies rise beyond 5, though extreme cases with very high revol_util or DTI still exist. The bar plots illustrate the upward trend in default risk for moderate past delinquencies, while the pie chart highlights that the majority of defaults cluster in borrowers with 3–4 prior delinquencies. Overall, past delinquencies, combined with high revol_util and DTI, serve as strong compounded risk indicators for future loan performance.

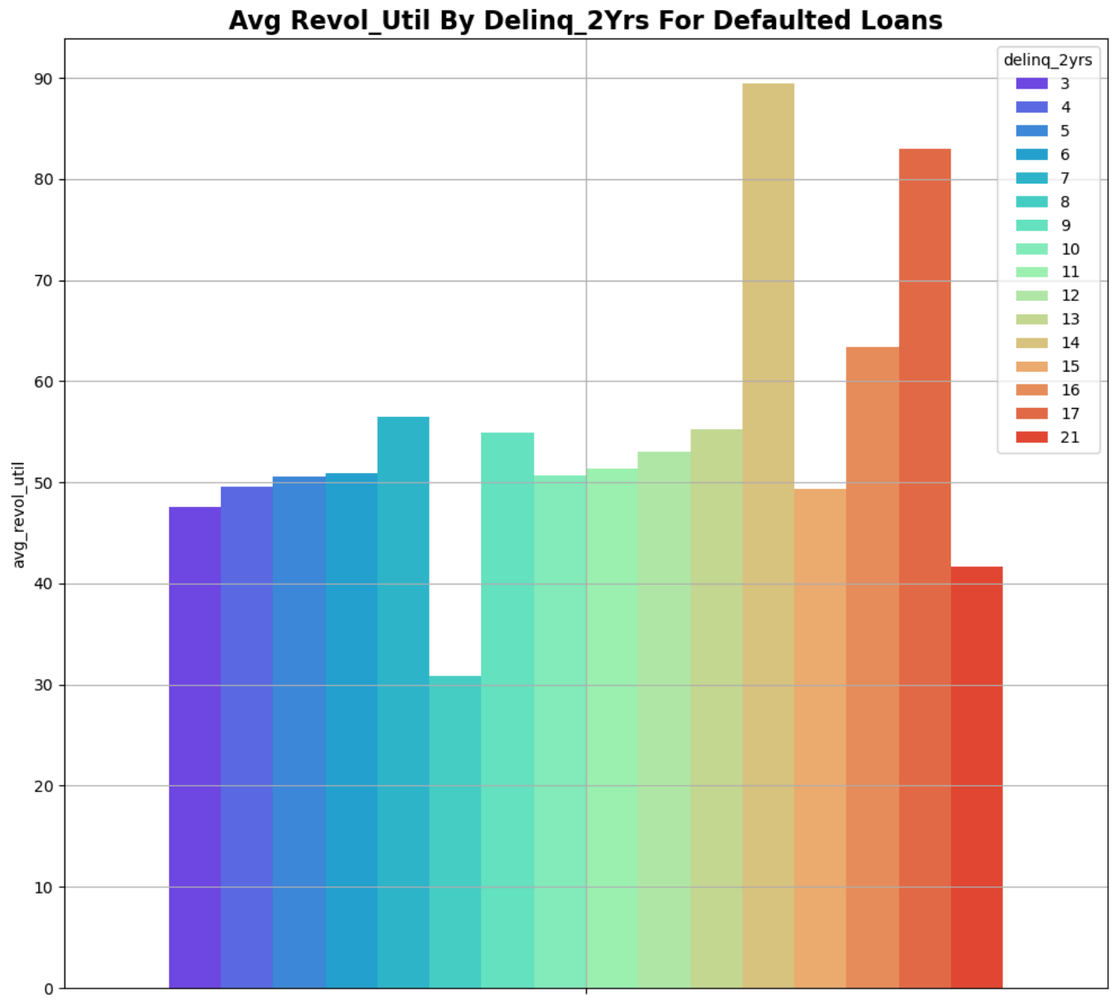
In [62]:
```
query = """SELECT delinq_2yrs,ROUND(AVG(revol_util),2) AS avg_revol_util,ROUND(AVg(dti),2) AS avg_dti,COUNT(loan_id) AS 'num_loans',
           ROUND(COUNT(loan_id)  * 100 / (SELECT COUNT(*) FROM loans WHERE loan_status = 'charged off'AND delinq_2yrs >2),2)
           AS 'percent_charged_off_loans'
           FROM loans
           WHERE loan_status = 'charged off' AND delinq_2yrs > 2
           GROUP BY delinq_2yrs
           ORDER BY delinq_2yrs;"""

df = run_query(conn,query)
df['delinq_2yrs'] = df['delinq_2yrs'].astype('category')
df['avg_revol_util'] = df['avg_revol_util'].astype('float')
df['avg_dti'] = df['avg_dti'].astype('float')
df['num_loans'] = df['num_loans'].astype('int')
df['percent_charged_off_loans'] = df['percent_charged_off_loans'].astype('float')
df
```

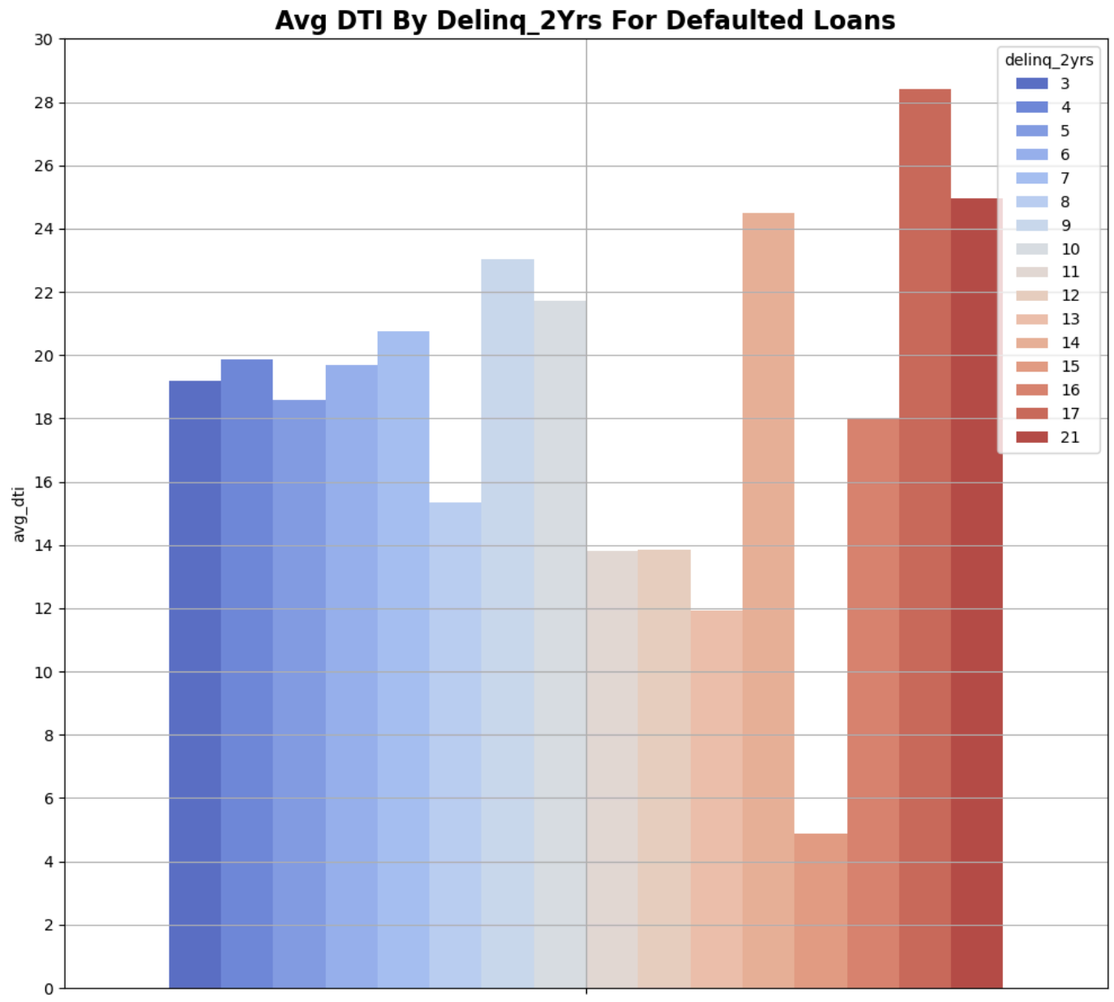
Out[62]:

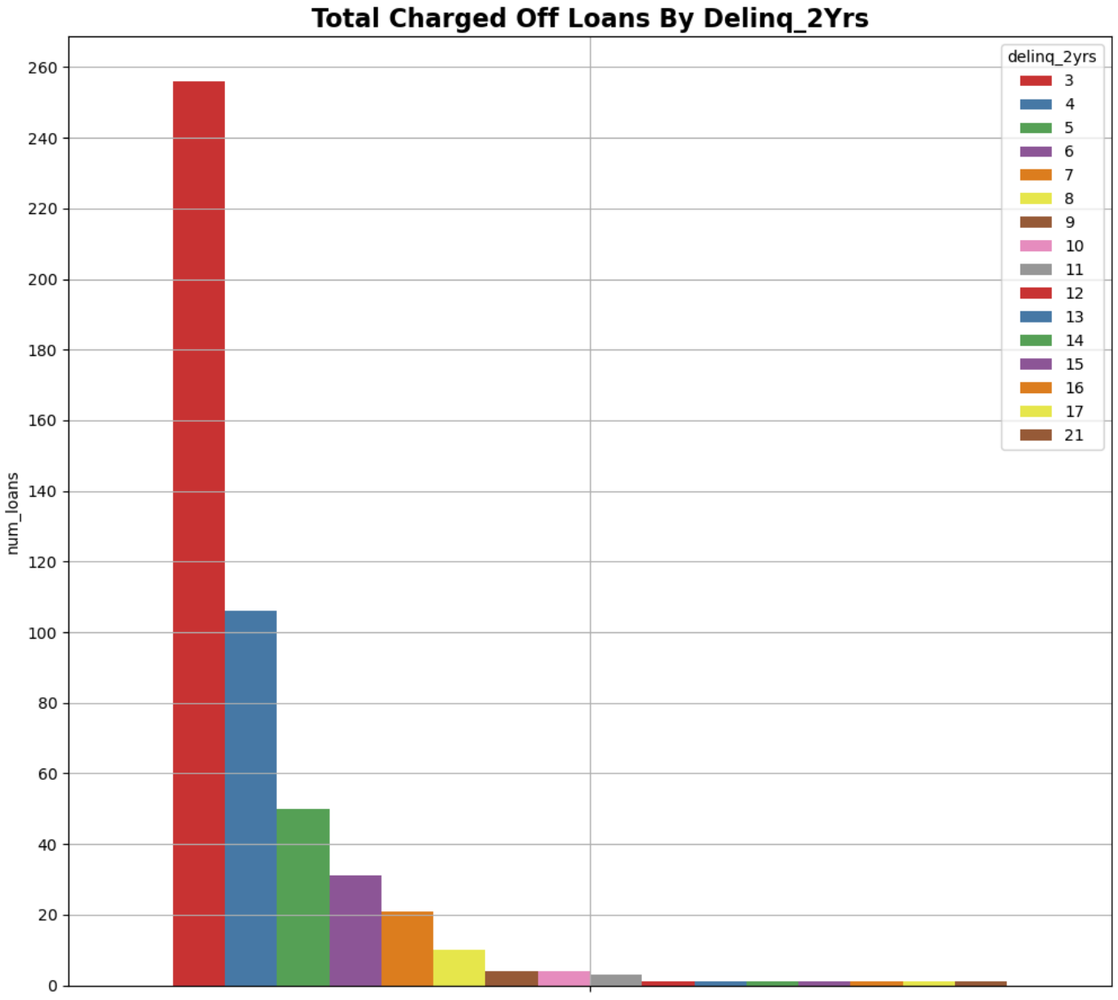
	delinq_2yrs	avg_revol_util	avg_dti	num_loans	percent_charged_off_loans
0	3	47.56	19.19	256	52.03
1	4	49.56	19.85	106	21.54
2	5	50.61	18.58	50	10.16
3	6	50.89	19.69	31	6.30
4	7	56.46	20.75	21	4.27
5	8	30.91	15.34	10	2.03
6	9	54.88	23.04	4	0.81
7	10	50.73	21.70	4	0.81
8	11	51.37	13.81	3	0.61
9	12	53.00	13.84	1	0.20
10	13	55.20	11.91	1	0.20
11	14	89.40	24.48	1	0.20
12	15	49.40	4.86	1	0.20
13	16	63.40	18.02	1	0.20
14	17	83.00	28.41	1	0.20
15	21	41.70	24.97	1	0.20

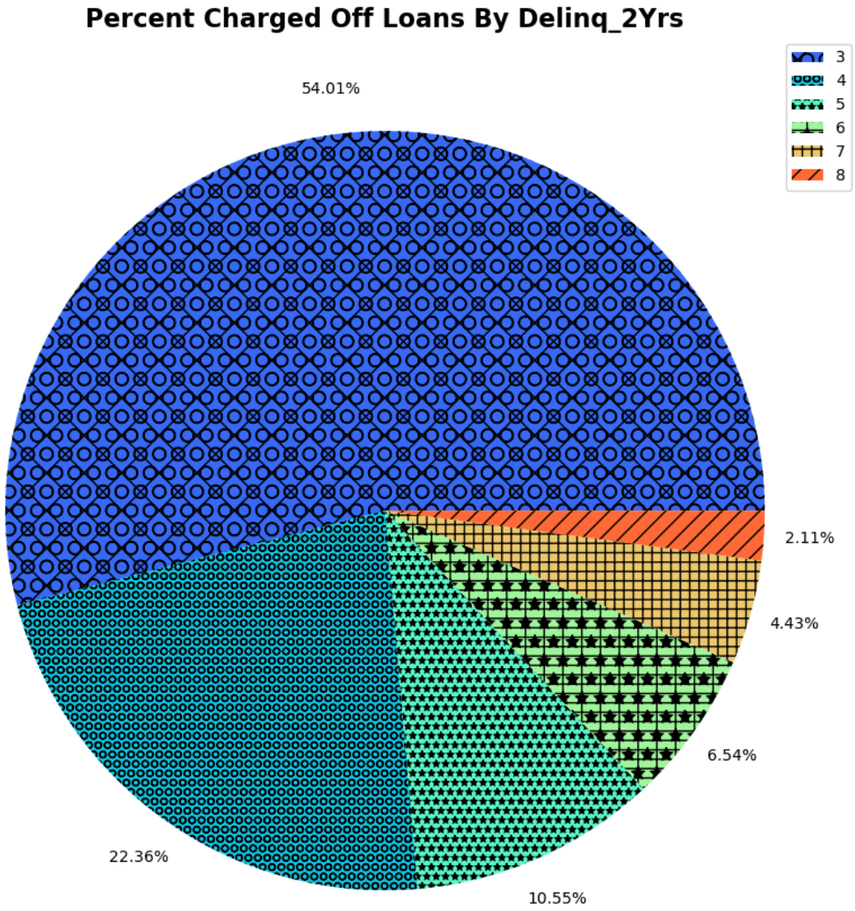
In [63]:
```
fig,ax = plt.subplots(nrows = 2,ncols = 2,figsize = (20,18))

sns.barplot(data = df,hue = 'delinq_2yrs',y = 'avg_revol_util',palette = 'rainbow',ax = ax[0,0])
ax[0,0].grid(True)
ax[0,0].set_yticks(ticks = range(0,100,10))
ax[0,0].set_title('Avg Revol_Util By Delinq_2Yrs For Defaulted Loans',fontweight = 'bold',fontsize = '16')

sns.barplot(data = df,hue = 'delinq_2yrs',y = 'avg_dti',palette = 'coolwarm',ax = ax[0,1])
ax[0,1].grid(True)
ax[0,1].set_yticks(ticks = range(0,32,2))
ax[0,1].set_title('Avg DTI By Delinq_2Yrs For Defaulted Loans',fontweight = 'bold',fontsize = '16')

sns.barplot(data = df,hue = 'delinq_2yrs',y = 'num_loans',palette = 'Set1',ax = ax[1,0])
ax[1,0].grid(True)
ax[1,0].set_yticks(ticks = range(0,280,20))
ax[1,0].set_title('Total Charged Off Loans By Delinq_2Yrs',fontweight = 'bold',fontsize = '16')

ax[1,1].pie(df['percent_charged_off_loans'].head(6),autopct = '%1.2f%%',hatch = ['xO','OO','**','*+-','++','//'],pctdistance = 1.12,
            colors = sns.color_palette('rainbow'))
ax[1,1].legend(labels = df['delinq_2yrs'],loc = 'best')
ax[1,1].set_title('Percent Charged Off Loans By Delinq_2Yrs',fontweight = 'bold',fontsize = '16')
```

```
plt.tight_layout()
plt.show()
```









22. Income Bands vs Default % along with dti and total_acc. Links economic background with repayment behavior; higher leverage or more accounts may compound risk.

The Income Bands vs Default % analysis shows that default rates are surprisingly similar across all income levels, hovering around 19–20%, suggesting that income alone isn't a strong predictor of default. Average DTI remains consistently high (~21–22%) across bands, while total accounts vary more, with ultra-high and high-income borrowers holding 27–29 accounts on average, compared to 19–25 for lower-income bands, indicating higher financial activity. The bar plots illustrate that defaults are evenly distributed across income segments, and DTI does not drastically differ, whereas the total number of accounts grows with income. The pie chart confirms that each income band contributes similarly to overall defaults. Overall, repayment risk appears relatively uniform across economic strata, but higher leverage or more accounts may subtly compound the risk for wealthier borrowers.

In [64]:
```
query = """ SELECT T1.income_band, ROUND(AVG(T1.annual_inc),2) AS 'avg_annual_inc',
            ROUND(AVG(T2.dti),2) AS 'avg_dti',
            ROUND(AVG(T1.total_acc),2) AS 'avg_total_acc',
            COUNT(T2.loan_id) AS 'num_loans',
            ROUND(COUNT(T2.loan_id) * 100 /(SELECT COUNT(loan_id) FROM loans WHERE loan_status = 'charged off'),2) AS 'percent_default_loans'
            FROM (SELECT*, CASE
                            WHEN t.bucket = 1 THEN 'low'
                            WHEN t.bucket = 2 THEN 'lower_middle'
                            WHEN t.bucket = 3 THEN 'middle'
                            WHEN t.bucket = 4 THEN 'high'
                            WHEN t.bucket = 5 THEN 'ultra_high'
                        END AS 'income_band'
                FROM (SELECT *, NTILE(5) OVER(ORDER BY annual_inc) AS 'bucket' FROM borrowers) t) T1
            JOIN loans T2
            ON T1.borrower_id = T2.borrower_id
            WHERE T2.loan_status = 'charged off'
            GROUP BY T1.income_band
            ORDER BY num_loans DESC;"""

df = run_query(conn,query)
df
```

Out[64]:

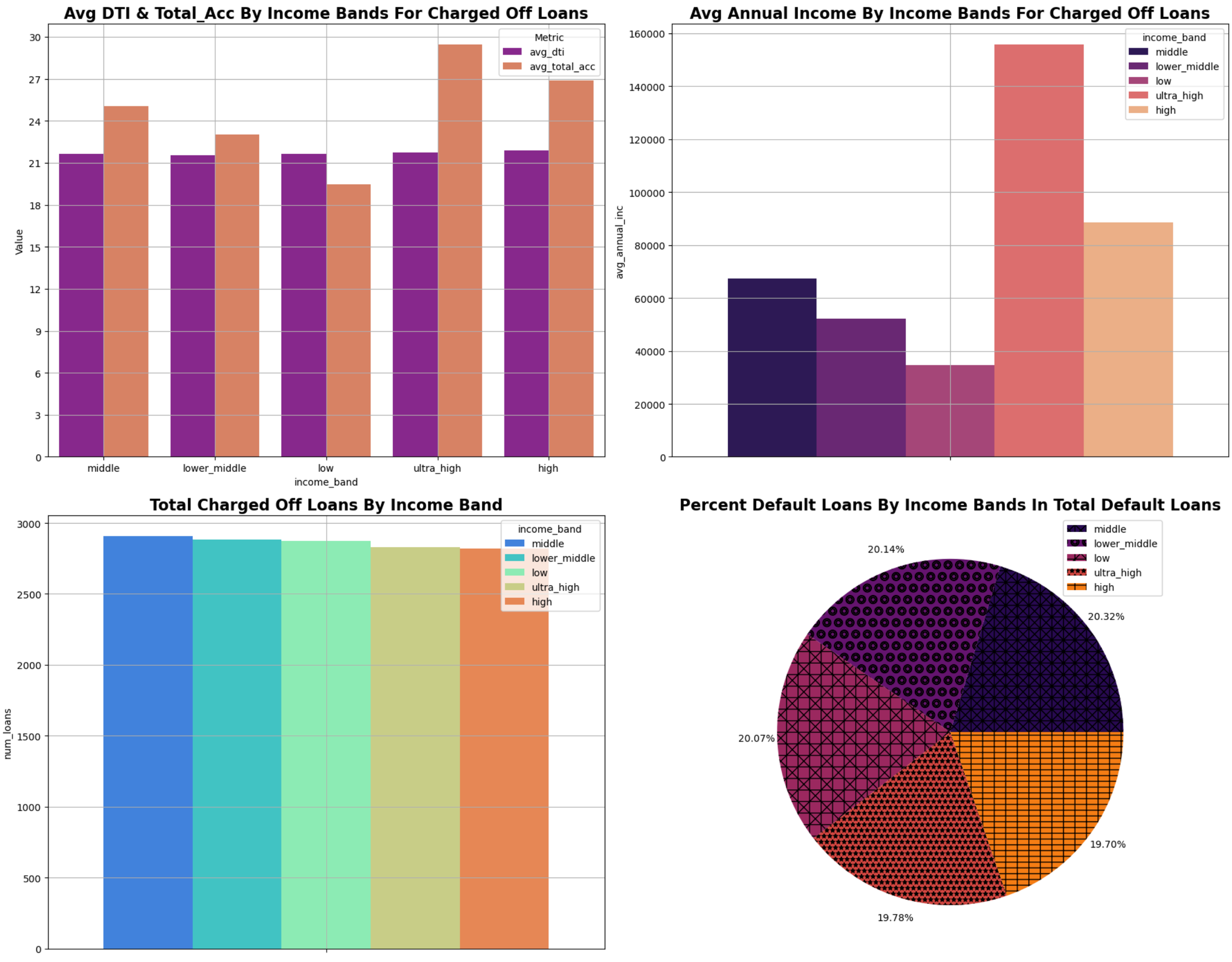
	income_band	avg_annual_inc	avg_dti	avg_total_acc	num_loans	percent_default_loans
0	middle	67369.62	21.62	25.07	2907	20.32
1	lower_middle	52056.52	21.52	23.02	2881	20.14
2	low	34723.61	21.65	19.47	2872	20.07
3	ultra_high	155747.09	21.74	29.45	2830	19.78
4	high	88422.02	21.88	26.87	2818	19.70

In [65]:
```
df_melt = df.melt(
    id_vars = ['income_band'],
    value_vars = ['avg_dti', 'avg_total_acc'],
    var_name = 'Metric',
    value_name = 'Value'
)

df_melt
```

Out[65]:

	income_band	Metric	Value
0	middle	avg_dti	21.62
1	lower_middle	avg_dti	21.52
2	low	avg_dti	21.65
3	ultra_high	avg_dti	21.74
4	high	avg_dti	21.88
5	middle	avg_total_acc	25.07
6	lower_middle	avg_total_acc	23.02
7	low	avg_total_acc	19.47
8	ultra_high	avg_total_acc	29.45
9	high	avg_total_acc	26.87

In [66]:
```
fig,ax = plt.subplots(nrows = 2,ncols = 2,figsize = (18,14))

sns.barplot(data = df_melt,hue = 'Metric',x = 'income_band',y = 'Value',palette = 'plasma',ax = ax[0,0])
ax[0,0].set_title('Avg DTI & Total_Acc By Income Bands For Charged Off Loans',fontweight = 'bold',fontsize = '16')
ax[0,0].grid(True)
ax[0,0].set_yticks(ticks = range(0,33,3))

sns.barplot(data = df,hue = 'income_band',y = 'avg_annual_inc',palette = 'magma',ax = ax[0,1])
ax[0,1].set_title('Avg Annual Income By Income Bands For Charged Off Loans',fontweight = 'bold',fontsize = '16')
ax[0,1].grid(True)

sns.barplot(data = df,hue = 'income_band',y = 'num_loans',palette = 'rainbow',ax = ax[1,0])
ax[1,0].set_title('Total Charged Off Loans By Income Band',fontweight = 'bold',fontsize = '16')
ax[1,0].grid(True)

ax[1,1].pie(df['percent_default_loans'],autopct = '%1.2f%%',hatch = ['Xx+','oO','+X','**','-+'],pctdistance = 1.12,
            colors = sns.color_palette('inferno'))
ax[1,1].legend(labels = df['income_band'])
ax[1,1].set_title('Percent Default Loans By Income Bands In Total Default Loans',fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```



23. Interest Rate Bands vs Default %. Verifies correlation of high interest rates with poor repayment while accounting for grading and loan duration.

The Interest Rate Bands vs Default % analysis highlights that higher interest rates are strongly associated with greater default risk, particularly among lower grades and longer-term loans. Loans with rates above 20% (mostly grades E, F, G) show both higher absolute defaults and higher default percentages, while A and B-grade loans at lower rates (<15%) exhibit minimal defaults. Medium-risk borrowers (grades C and D) see rising defaults as rates move from 10–15% to 15–20%, showing a clear sensitivity to interest

costs. The stacked bar plots illustrate how defaults concentrate in high-rate bands and lower grades, and the percentage plot confirms that high-rate, long-term loans disproportionately contribute to overall charged-off loans. Overall, interest rate is a key driver of repayment risk, especially for weaker credit grades and longer durations.

In [67]:
```
query = """WITH T AS (SELECT *,
                              CASE
                                  WHEN int_rate < 10 THEN '<10%'
                                  WHEN int_rate BETWEEN 10 AND 15 THEN '10-15%'
                                  WHEN int_rate BETWEEN 15 AND 20 THEN '15-20%'
                                  WHEN int_rate > 20 THEN '>20%'
                              END AS 'int_rate_range'
                      FROM loans)


          SELECT grade,term,int_rate_range,COUNT(loan_id) AS 'total_default_loans',
          ROUND(COUNT(loan_id) * 100 / (SELECT COUNT(loan_id) FROM loans WHERE loan_status = 'charged off'),2)
          AS 'percent_default_loans'
          FROM T
          WHERE loan_status = 'charged off'
          GROUP BY int_rate_range,grade,term
          ORDER BY grade,term;"""

df = run_query(conn,query)

pivot_df1 = df.pivot(index = 'grade',
                       columns = ["int_rate_range","term"],
                       values = 'total_default_loans').fillna(0)
pivot_df1 = pivot_df1.astype('float')


pivot_df2 = df.pivot(index = 'grade',
                       columns = ["int_rate_range","term"],
                       values = 'percent_default_loans').fillna(0)
pivot_df2 = pivot_df2.astype('float')
```

In [68]:
```
pivot_df1
```

Out[68]:

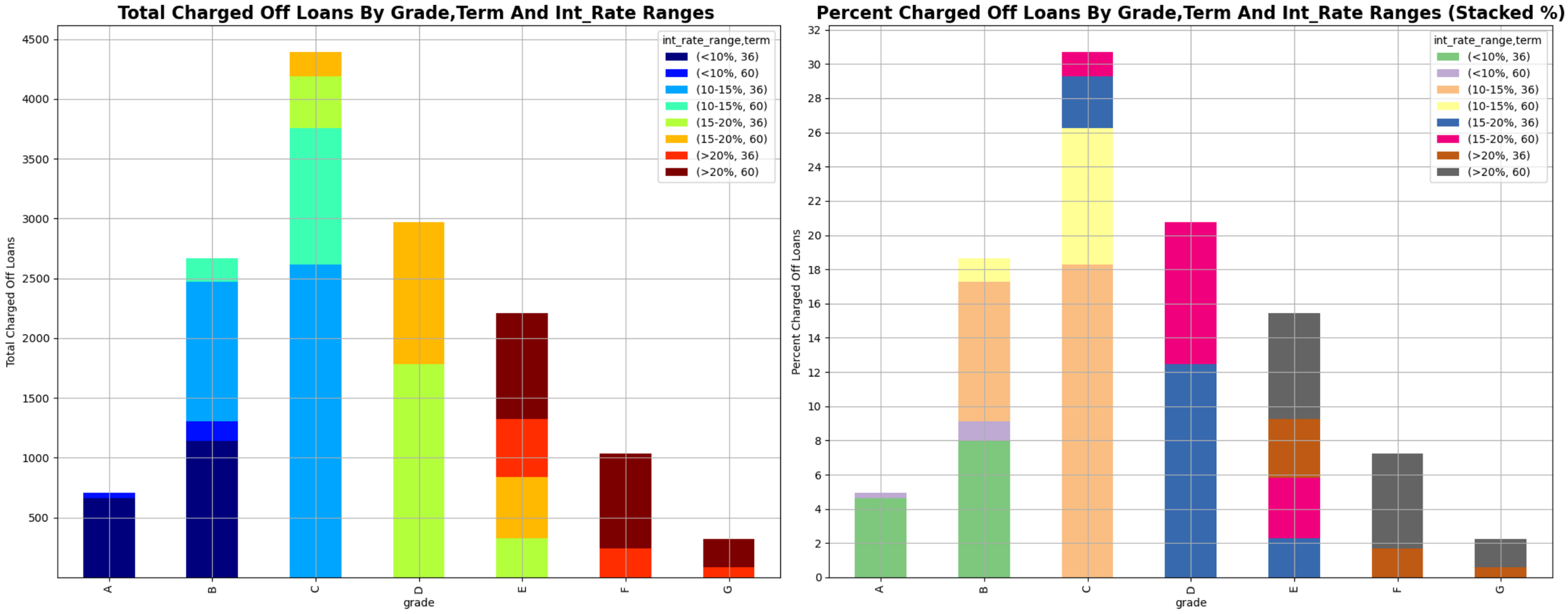
int_rate_range	<10%		10-15%		15-20%		>20%	
term	36	60	36	60	36	60	36	60
grade								
A	661.0	50.0	0.0	0.0	0.0	0.0	0.0	0.0
B	1139.0	165.0	1168.0	198.0	0.0	0.0	0.0	0.0
C	1.0	0.0	2615.0	1143.0	427.0	207.0	0.0	0.0
D	0.0	0.0	1.0	0.0	1783.0	1185.0	0.0	0.0
E	0.0	0.0	0.0	0.0	328.0	508.0	486.0	887.0
F	0.0	0.0	0.0	0.0	0.0	0.0	241.0	794.0
G	0.0	1.0	0.0	0.0	0.0	0.0	81.0	239.0

In [69]:
```
pivot_df2
```

Out[69]:

int_rate_range	<10%		10-15%		15-20%		>20%	
term	36	60	36	60	36	60	36	60
grade								
A	4.62	0.35	0.00	0.00	0.00	0.00	0.00	0.00
B	7.96	1.15	8.16	1.38	0.00	0.00	0.00	0.00
C	0.01	0.00	18.28	7.99	2.98	1.45	0.00	0.00
D	0.00	0.00	0.01	0.00	12.46	8.28	0.00	0.00
E	0.00	0.00	0.00	0.00	2.29	3.55	3.40	6.20
F	0.00	0.00	0.00	0.00	0.00	0.00	1.68	5.55
G	0.00	0.01	0.00	0.00	0.00	0.00	0.57	1.67

In [70]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (20,8))

ax[0] = pivot_df1.plot(kind = "bar", stacked = True, colormap = 'jet',ax = ax[0])
ax[0].grid(True)
ax[0].set_ylabel("Total Charged Off Loans")
ax[0].set_yticks(ticks = range(500,5000,500))
ax[0].set_title("Total Charged Off Loans By Grade,Term And Int_Rate Ranges ",fontweight = 'bold',fontsize = '16')

ax[1] = pivot_df2.plot(kind = "bar", stacked = True, colormap = 'Accent',ax = ax[1])
ax[1].grid(True)
ax[1].set_yticks(ticks = range(0,34,2))
ax[1].set_ylabel("Percent Charged Off Loans")
ax[1].set_title("Percent Charged Off Loans By Grade,Term And Int_Rate Ranges (Stacked %)",fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```



24. Grade Migration Analysis. Track borrowers' grade changes between consecutive loans to find cases where grade worsened. Identifies borrowers whose creditworthiness declined over time; helps lenders flag potential future defaults.

The Grade Migration Analysis reveals how borrower's creditworthiness evolves across loans. Most high-grade borrowers (A and B) tend to maintain or improve their grades, while lower grades (C–G) show significant worsening, indicating rising risk. The stacked bar shows that No Change dominates, but Worsened loans increase sharply in grades C, D, and E. The histogram of grade changes confirms most changes are small, though negative changes are concentrated in mid-lower grades. The heatmap of previous vs. current grades highlights common transitions, showing some borrowers drop multiple grades between loans, and the boxplot illustrates the spread and outliers of grade changes per current grade. Overall, tracking grade migration helps lenders flag borrowers whose risk profile is deteriorating over time.

In [71]:
```
query = """WITH grade_mapped AS (SELECT borrower_id,loan_id, grade,loan_status,
                                  CASE grade
                                    WHEN 'A' THEN 1
                                    WHEN 'B' THEN 2
                                    WHEN 'C' THEN 3
                                    WHEN 'D' THEN 4
                                    WHEN 'E' THEN 5
                                    WHEN 'F' THEN 6
                                    WHEN 'G' THEN 7
                                  END AS grade_num
                                  FROM loans)

SELECT borrower_id,loan_id,loan_status,grade AS current_grade,grade_num,
       LAG(grade) OVER (PARTITION BY borrower_id ORDER BY loan_id) AS prev_grade,
       LAG(grade_num) OVER (PARTITION BY borrower_id ORDER BY loan_id) AS grade_num_change,
       grade_num - LAG(grade_num) OVER (PARTITION BY borrower_id ORDER BY loan_id) AS grade_change,
       CASE
          WHEN grade_num > LAG(grade_num) OVER (PARTITION BY borrower_id ORDER BY loan_id) THEN 'Worsened'
          WHEN grade_num < LAG(grade_num) OVER (PARTITION BY borrower_id ORDER BY loan_id) THEN 'Improved'
          ELSE 'No Change'
      END AS grade_migration
FROM grade_mapped
ORDER BY borrower_id, loan_id;"""

df = run_query(conn,query)
df['grade_change'] = df['grade_change'].fillna(0).astype('int')
df
```

Out[71]:

	borrower_id	loan_id	loan_status	current_grade	grade_num	prev_grade	grade_num_change	grade_change	grade_migration
0	1	LN000001	late	B	2	None	NaN	0	No Change
1	1	LN000002	current	C	3	B	2.0	1	Worsened
2	1	LN000003	current	C	3	C	3.0	0	No Change
3	2	LN000004	current	B	2	None	NaN	0	No Change
4	2	LN000005	current	A	1	B	2.0	-1	Improved
...	...	...	...	...	...	...	...	...	...
124870	49949	LN124939	current	C	3	None	NaN	0	No Change
124871	49949	LN124940	fully paid	A	1	C	3.0	-2	Improved
124872	49949	LN124941	current	E	5	A	1.0	4	Worsened
124873	49950	LN124942	current	B	2	None	NaN	0	No Change
124874	49950	LN124943	current	A	1	B	2.0	-1	Improved

124875 rows × 9 columns

In [72]:
```
pivot_df1 = df.pivot_table(index = "current_grade", columns = "grade_migration", values = "loan_id", aggfunc = "count", fill_value = 0)
pivot_df1
```

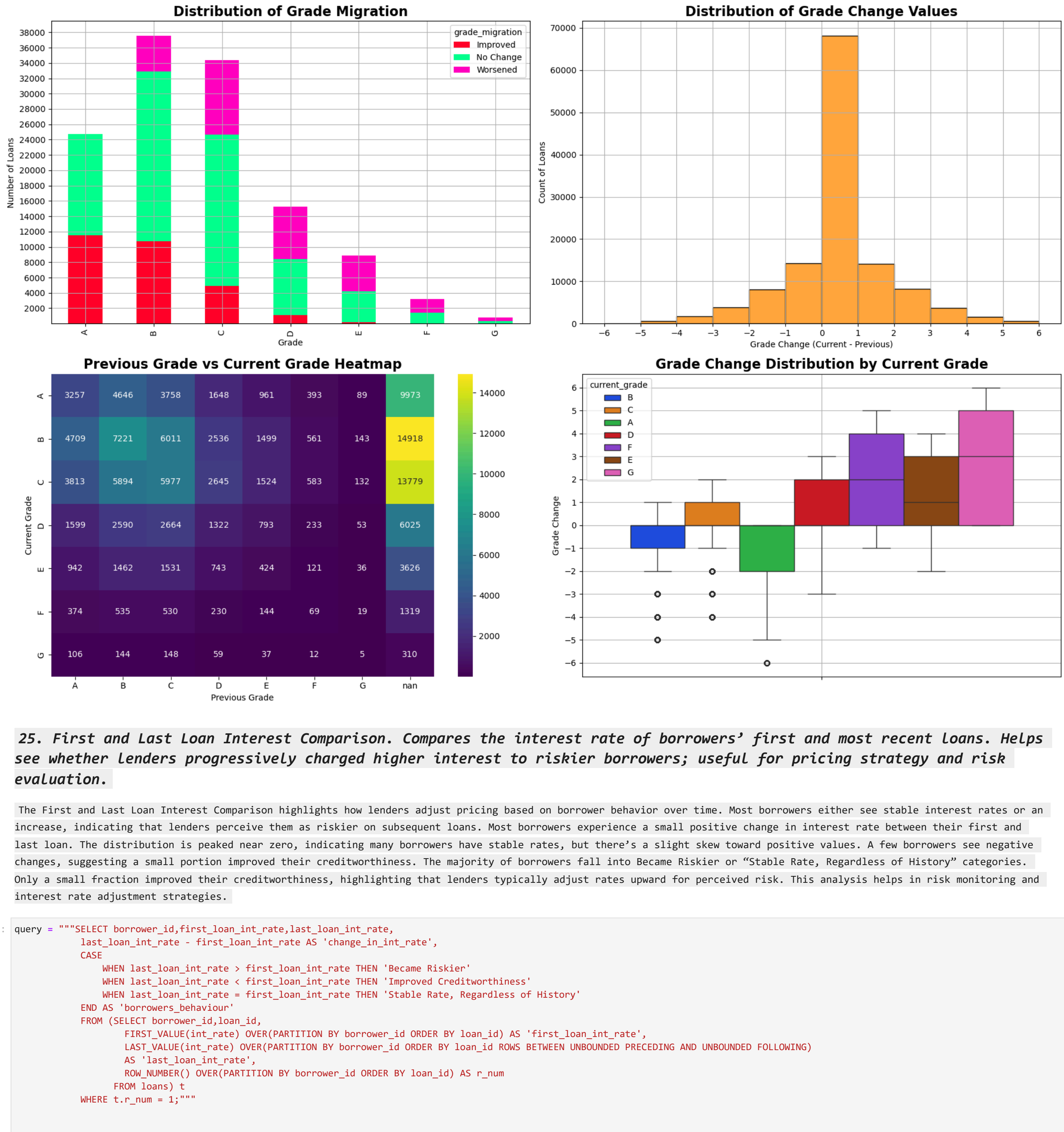
Out[72]:

grade_migration	Improved	No Change	Worsened
current_grade			
A	11495	13230	0
B	10750	22139	4709
C	4884	19756	9707
D	1079	7347	6853
E	157	4050	4678
F	19	1388	1813
G	0	315	506

In [73]:
```
fig,ax = plt.subplots(nrows = 2,ncols = 2,figsize = (18,12))

# Shows how many loans improved, worsened, or stayed the same per grade.
ax[0,0] = pivot_df1.plot(kind = 'bar',stacked = True,colormap = 'gist_rainbow',ax = ax[0,0])
ax[0,0].set_title('Distribution of Grade Migration', fontweight = 'bold',fontsize = '16')
ax[0,0].set_ylabel('Number of Loans')
ax[0,0].set_xlabel('Grade')
```

```
ax[0,0].set_yticks(ticks = range(2000,40000,2000))
ax[0,0].grid(True)

# Grade Change Histogram
sns.histplot(data = df, x = 'grade_change', bins = np.arange(-6,7,1), color = 'darkorange',ax = ax[0,1])
ax[0,1].set_title('Distribution of Grade Change Values', fontweight = 'bold',fontsize = '16')
ax[0,1].set_xlabel('Grade Change (Current - Previous)')
ax[0,1].set_ylabel('Count of Loans')
ax[0,1].grid(True)
ax[0,1].set_xticks(ticks = range(df['grade_change'].min(),df['grade_change'].max()+1))

# Highlights common grade migrations (who worsens/improves/stays same).
migration_matrix = pd.crosstab(df['current_grade'],df['prev_grade'],dropna = False)
print(migration_matrix,end = '\n\n')

sns.heatmap(migration_matrix, annot = True, fmt = 'd', cmap = 'viridis',ax = ax[1,0])
ax[1,0].set_title('Previous Grade vs Current Grade Heatmap', fontweight = 'bold',fontsize = '16')
ax[1,0].set_xlabel('Previous Grade')
ax[1,0].set_ylabel('Current Grade')

# Shows spread of grade change values per current grade and identifies outliers.
sns.boxplot(data = df, hue = 'current_grade', y = 'grade_change', palette = 'bright',ax = ax[1,1])
ax[1,1].set_title('Grade Change Distribution by Current Grade', fontweight = 'bold',fontsize = '16')
ax[1,1].set_ylabel('Grade Change')
ax[1,1].set_yticks(ticks = range(-6,7))
ax[1,1].grid(True)

plt.tight_layout()
plt.show()
```

```
prev_grade         A     B     C     D     E    F    G   NaN
current_grade
A               3257  4646  3758  1648   961  393   89  9973
B               4709  7221  6011  2536  1499  561  143 14918
C               3813  5894  5977  2645  1524  583  132 13779
D               1599  2590  2664  1322   793  233   53  6025
E                942  1462  1531   743   424  121   36  3626
F                374   535   530   230   144   69   19  1319
G                106   144   148    59    37   12    5   310
```



25. First and Last Loan Interest Comparison. Compares the interest rate of borrowers' first and most recent loans. Helps see whether lenders progressively charged higher interest to riskier borrowers; useful for pricing strategy and risk evaluation.

The First and Last Loan Interest Comparison highlights how lenders adjust pricing based on borrower behavior over time. Most borrowers either see stable interest rates or an increase, indicating that lenders perceive them as riskier on subsequent loans. Most borrowers experience a small positive change in interest rate between their first and last loan. The distribution is peaked near zero, indicating many borrowers have stable rates, but there's a slight skew toward positive values. A few borrowers see negative changes, suggesting a small portion improved their creditworthiness. The majority of borrowers fall into Became Riskier or "Stable Rate, Regardless of History" categories. Only a small fraction improved their creditworthiness, highlighting that lenders typically adjust rates upward for perceived risk. This analysis helps in risk monitoring and interest rate adjustment strategies.

In [74]:
```
query = """SELECT borrower_id,first_loan_int_rate,last_loan_int_rate,
           last_loan_int_rate - first_loan_int_rate AS 'change_in_int_rate',
           CASE
               WHEN last_loan_int_rate > first_loan_int_rate THEN 'Became Riskier'
               WHEN last_loan_int_rate < first_loan_int_rate THEN 'Improved Creditworthiness'
               WHEN last_loan_int_rate = first_loan_int_rate THEN 'Stable Rate, Regardless of History'
           END AS 'borrowers_behaviour'
           FROM (SELECT borrower_id,loan_id,
                   FIRST_VALUE(int_rate) OVER(PARTITION BY borrower_id ORDER BY loan_id) AS 'first_loan_int_rate',
                   LAST_VALUE(int_rate) OVER(PARTITION BY borrower_id ORDER BY loan_id ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
                   AS 'last_loan_int_rate',
                   ROW_NUMBER() OVER(PARTITION BY borrower_id ORDER BY loan_id) AS r_num
                 FROM loans) t
           WHERE t.r_num = 1;"""
```

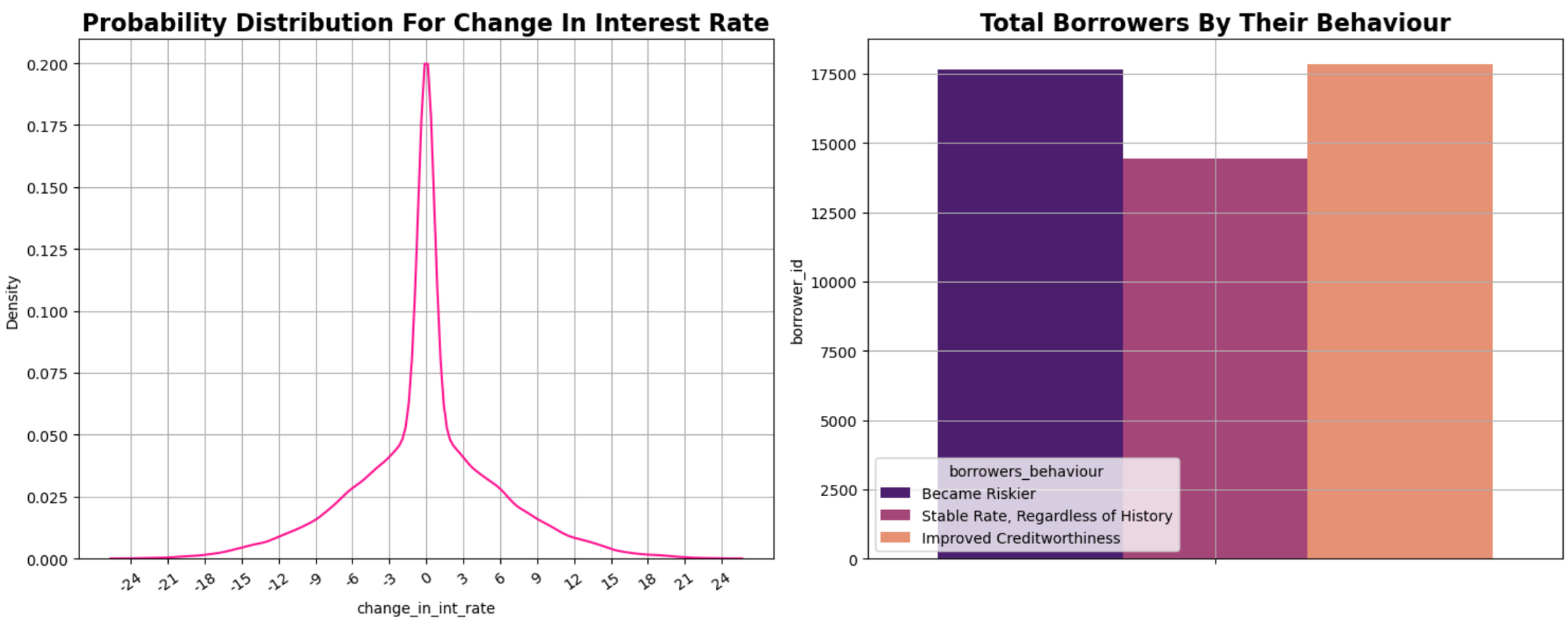
```
df = run_query(conn,query)
df
```

Out[74]:

	borrower_id	first_loan_int_rate	last_loan_int_rate	change_in_int_rate	borrowers_behaviour
0	1	9.16	14.46	5.30	Became Riskier
1	2	9.16	18.99	9.83	Became Riskier
2	3	26.57	26.57	0.00	Stable Rate, Regardless of History
3	4	17.27	22.45	5.18	Became Riskier
4	5	11.99	11.99	0.00	Stable Rate, Regardless of History
...	...	...	...	...	...
49945	49946	12.59	12.59	0.00	Stable Rate, Regardless of History
49946	49947	10.64	10.64	0.00	Stable Rate, Regardless of History
49947	49948	11.22	14.48	3.26	Became Riskier
49948	49949	11.99	19.99	8.00	Became Riskier
49949	49950	9.17	7.89	-1.28	Improved Creditworthiness

49950 rows × 5 columns

In [75]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (15,6))
sns.kdeplot(data = df,x = 'change_in_int_rate',ax = ax[0],color = 'deeppink')
ax[0].set_xticks(ticks = range(-24,27,3))
ax[0].set_xticklabels(labels = range(-24,27,3),rotation = 35)
ax[0].set_title('Probability Distribution For Change In Interest Rate',fontsize = '16',fontweight = 'bold')
ax[0].grid(True)

sns.barplot(data = df,hue = 'borrowers_behaviour',y = 'borrower_id',estimator = np.count_nonzero,palette = 'magma',ax = ax[1])
ax[1].set_title('Total Borrowers By Their Behaviour',fontsize = '16',fontweight = 'bold')
ax[1].grid(True)

plt.tight_layout()
plt.show()
```



In [ ]:

In [ ]:

D. Borrower Behavior & Repayment: How do borrower traits influence repayment?

Borrowers' traits and financial habits strongly influence repayment. In this section, we analyze how factors like employment, housing, income, and verification affect loan outcomes.

Summary-

A borrower's personal traits and financial habits play a big role in whether they repay loans on time. People with stable jobs, longer work history, and owned or mortgaged homes tend to default less because they have more financial stability. Renters or those with short employment histories are more likely to default, especially on mid-risk loans, showing that instability increases repayment risk. Repeat defaulters, particularly those with many accounts or multiple active loans, show a pattern of financial stress and difficulty managing debt. Even high-income borrowers are not immune—verification alone does not prevent defaults. Mid-risk loan grades see higher defaults even among verified or wealthy borrowers, showing that loan quality and risk profile matter more than income verification alone.

Past loan behavior and borrowing patterns also strongly predict future repayment. Borrowers whose first loan was charged off often default again, while those who fully repaid or stayed current are more likely to maintain good repayment habits. Many repeat defaulters take larger loans over time, which increases their default risk, while some take smaller loans, possibly due to stricter lending limits. Even though income, interest rates, and debt-to-income ratios influence repayment, they are not enough to fully predict defaults. Overall, lenders should combine information on employment stability, housing, loan history, borrowing patterns, and financial stress to identify high-risk borrowers early and manage their loan portfolios more effectively.

In [ ]:

26. Multi-Dimensional Borrower Risk. Calculates default % for borrowers segmented by (emp_length + home_ownership + grade). Shows combined effect of stability and loan quality on repayment risk; highly useful for portfolio risk assessment. Stable housing and employment usually indicate lower risk; combining them reveals stronger stability signals.

The multi-dimensional borrower risk analysis clearly shows that both financial stability and loan quality drive repayment outcomes. Borrowers with own house, mortgages and longer employment histories consistently exhibit lower default rates across nearly all grades, highlighting the protective effect of stable housing and work. Renters and those with shorter employment tenures face higher defaults, particularly in mid-grades C and D, indicating that instability compounds risk even for moderately rated loans. High-grade loans (A–B) remain low-risk across all segments, while lower grades (E–G) show elevated defaults, especially among less stable borrowers. The stacked bar plots vividly illustrate how defaults concentrate in unstable borrower profiles, making hotspots easy to identify. By combining employment length, home ownership, and loan grade, lenders gain a nuanced view of risk, enabling more precise portfolio management. Overall, stability and loan quality together provide the strongest signals for predicting borrower repayment behavior.

In [153…
```
query = """ SELECT home_ownership,emp_length,grade,COUNT(loan_id) AS 'total_default_loans',
            ROUND(COUNT(loan_id) * 100 / (SELECT COUNT(loan_id) FROM loans WHERE loan_status = 'charged off'),2)
            AS 'percent_default_loans'
            FROM loans t1
            JOIN borrowers t2
            ON t1.borrower_id = t2.borrower_id
            WHERE t1.loan_status = 'charged off'
            GROUP BY t2.home_ownership,t2.emp_length,t1.grade
            ORDER BY home_ownership,emp_length DESC;"""

df = run_query(conn,query)
df
```

Out[153…

	home_ownership	emp_length	grade	total_default_loans	percent_default_loans
0	MORTGAGE	9	A	43	0.30
1	MORTGAGE	9	B	155	1.08
2	MORTGAGE	9	C	236	1.65
3	MORTGAGE	9	D	179	1.25
4	MORTGAGE	9	E	124	0.87
...	...	...	...	...	...
163	RENT	2	C	105	0.73
164	RENT	2	D	56	0.39
165	RENT	2	E	49	0.34
166	RENT	2	F	26	0.18
167	RENT	2	G	2	0.01

168 rows × 5 columns

In [154…
```
pivot_df1 = df.pivot_table(values = 'total_default_loans', index = ['home_ownership','emp_length'], columns = ['grade'])

pivot_df2 = df.pivot_table(values = 'percent_default_loans', index = ['home_ownership','emp_length'], columns = ['grade'])
```

In [155…
```
pivot_df1
```

Out[155…

	grade	A	B	C	D	E	F	G
home_ownership	**emp_length**							
MORTGAGE	**2**	12.0	45.0	82.0	53.0	44.0	22.0	1.0
	3	59.0	250.0	408.0	287.0	208.0	83.0	33.0
	4	44.0	184.0	265.0	179.0	137.0	83.0	28.0
	5	49.0	191.0	321.0	218.0	157.0	73.0	19.0
	6	31.0	151.0	246.0	161.0	122.0	63.0	22.0
	7	37.0	109.0	209.0	153.0	98.0	44.0	18.0
	8	52.0	160.0	292.0	228.0	156.0	76.0	21.0
	9	43.0	155.0	236.0	179.0	124.0	64.0	16.0
OWN	**2**	1.0	13.0	22.0	16.0	11.0	3.0	2.0
	3	15.0	48.0	104.0	64.0	51.0	28.0	7.0
	4	12.0	48.0	58.0	39.0	29.0	17.0	5.0
	5	11.0	49.0	78.0	54.0	41.0	13.0	8.0
	6	7.0	41.0	67.0	39.0	24.0	20.0	1.0
	7	6.0	24.0	39.0	32.0	23.0	10.0	5.0
	8	13.0	44.0	57.0	44.0	36.0	14.0	1.0
	9	8.0	39.0	61.0	44.0	25.0	11.0	2.0
RENT	**2**	14.0	76.0	105.0	56.0	49.0	26.0	2.0
	3	67.0	270.0	440.0	317.0	234.0	100.0	26.0
	4	51.0	149.0	282.0	178.0	131.0	68.0	18.0
	5	63.0	169.0	296.0	184.0	152.0	59.0	21.0
	6	31.0	141.0	205.0	129.0	83.0	45.0	22.0
	7	31.0	74.0	143.0	89.0	89.0	28.0	13.0
	8	30.0	139.0	191.0	126.0	103.0	41.0	17.0
	9	24.0	101.0	186.0	100.0	82.0	44.0	13.0

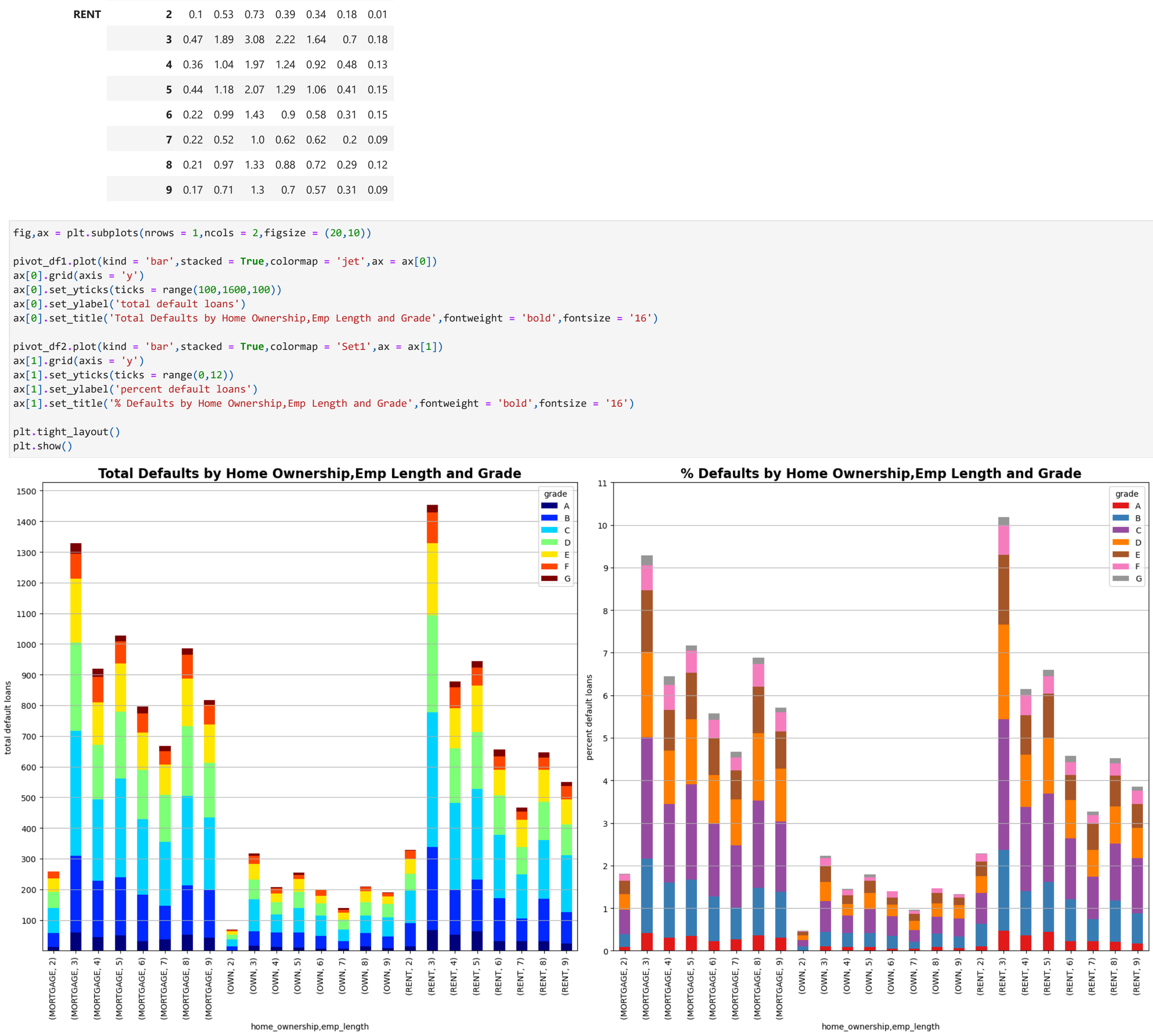
In [156…
```
pivot_df2
```

Out[156...

home_ownership	emp_length	grade A	B	C	D	E	F	G
MORTGAGE	**2**	0.08	0.31	0.57	0.37	0.31	0.15	0.01
	3	0.41	1.75	2.85	2.01	1.45	0.58	0.23
	4	0.31	1.29	1.85	1.25	0.96	0.58	0.2
	5	0.34	1.33	2.24	1.52	1.1	0.51	0.13
	6	0.22	1.06	1.72	1.13	0.85	0.44	0.15
	7	0.26	0.76	1.46	1.07	0.68	0.31	0.13
	8	0.36	1.12	2.04	1.59	1.09	0.53	0.15
	9	0.3	1.08	1.65	1.25	0.87	0.45	0.11
OWN	**2**	0.01	0.09	0.15	0.11	0.08	0.02	0.01
	3	0.1	0.34	0.73	0.45	0.36	0.2	0.05
	4	0.08	0.34	0.41	0.27	0.2	0.12	0.03
	5	0.08	0.34	0.55	0.38	0.29	0.09	0.06
	6	0.05	0.29	0.47	0.27	0.17	0.14	0.01
	7	0.04	0.17	0.27	0.22	0.16	0.07	0.03
	8	0.09	0.31	0.4	0.31	0.25	0.1	0.01
	9	0.06	0.27	0.43	0.31	0.17	0.08	0.01
RENT	**2**	0.1	0.53	0.73	0.39	0.34	0.18	0.01
	3	0.47	1.89	3.08	2.22	1.64	0.7	0.18
	4	0.36	1.04	1.97	1.24	0.92	0.48	0.13
	5	0.44	1.18	2.07	1.29	1.06	0.41	0.15
	6	0.22	0.99	1.43	0.9	0.58	0.31	0.15
	7	0.22	0.52	1.0	0.62	0.62	0.2	0.09
	8	0.21	0.97	1.33	0.88	0.72	0.29	0.12
	9	0.17	0.71	1.3	0.7	0.57	0.31	0.09

In [157...

```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (20,10))

pivot_df1.plot(kind = 'bar',stacked = True,colormap = 'jet',ax = ax[0])
ax[0].grid(axis = 'y')
ax[0].set_yticks(ticks = range(100,1600,100))
ax[0].set_ylabel('total default loans')
ax[0].set_title('Total Defaults by Home Ownership,Emp Length and Grade',fontweight = 'bold',fontsize = '16')

pivot_df2.plot(kind = 'bar',stacked = True,colormap = 'Set1',ax = ax[1])
ax[1].grid(axis = 'y')
ax[1].set_yticks(ticks = range(0,12))
ax[1].set_ylabel('percent default loans')
ax[1].set_title('% Defaults by Home Ownership,Emp Length and Grade',fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```



27. Repeat Defaulters and Loan Count. Finds borrowers who defaulted on multiple loans and have total_acc > 10. Identifies consistently risky individuals and those juggling many accounts.

The analysis of repeat defaulters highlights a small but significant segment of borrowers who default multiple times, despite holding numerous accounts. Borrowers with more than 10 total accounts and multiple defaults represent consistently high-risk individuals who may be over-leveraged or struggling to manage debt. Most repeat defaulters have 2–4 past defaults, showing a clear pattern of recurring repayment challenges. The bar plot illustrates that the majority of borrowers default only twice, signaling the need for closer monitoring. These high-risk borrowers are prime candidates for stricter credit assessment, targeted intervention, or adjusted lending terms. Overall, tracking repeat defaulters helps lenders identify persistent risk profiles and refine strategies to mitigate portfolio exposure.

In [81]:

```
query = """ SELECT t1.borrower_id,COUNT(t1.loan_id) AS 'num_defaulted_loans'
            FROM loans t1
            JOIN borrowers t2
            ON t1.borrower_id = t2.borrower_id
            WHERE t1.loan_status = 'charged off' AND t2.total_acc > 10
            GROUP BY t1.borrower_id
            HAVING num_defaulted_loans > 1
            ORDER BY num_defaulted_loans DESC;"""
```

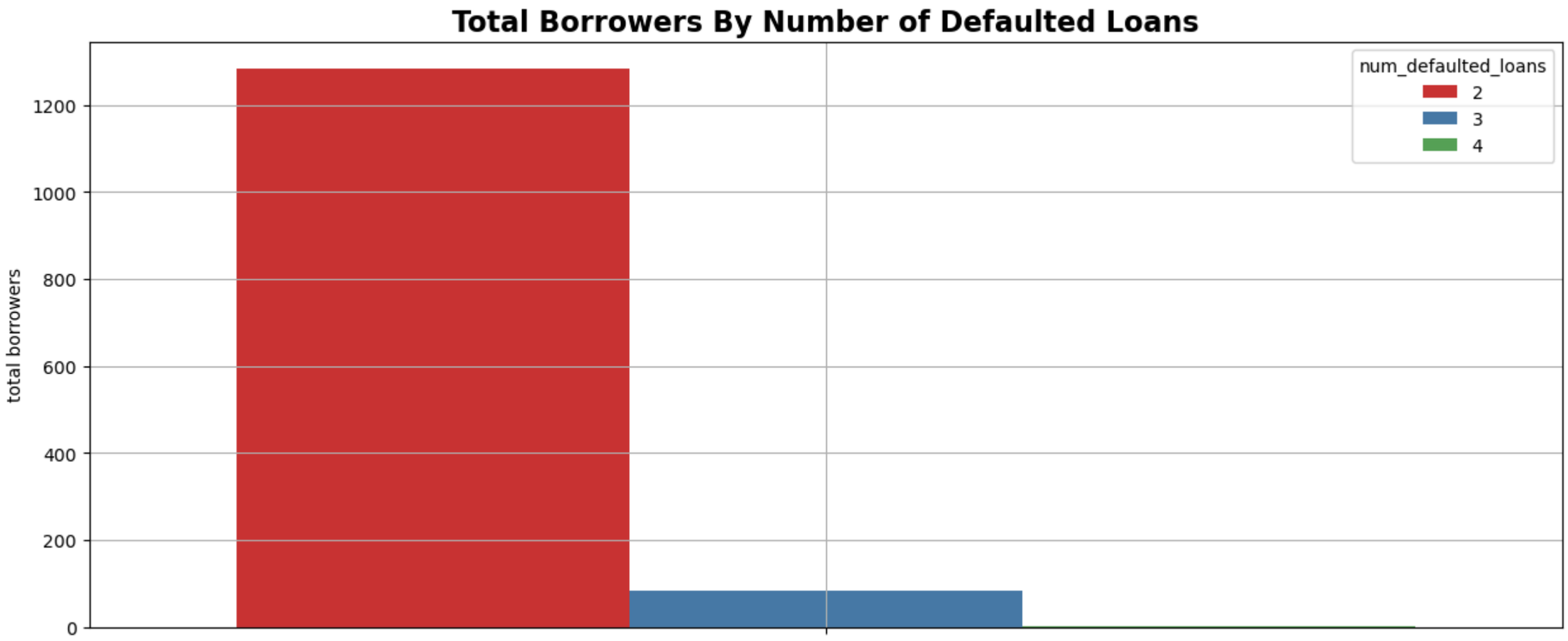
```
df = run_query(conn,query)
df
```

Out[81]:

	borrower_id	num_defaulted_loans
0	8235	4
1	561	3
2	806	3
3	819	3
4	896	3
...	...	...
1361	49696	2
1362	49711	2
1363	49760	2
1364	49770	2
1365	49888	2

1366 rows × 2 columns

In [82]:
```
plt.figure(figsize = (15,6))
sns.barplot(data = df,hue = 'num_defaulted_loans',y = 'borrower_id',estimator = np.count_nonzero,palette = 'Set1')
plt.grid(True)
plt.ylabel('total borrowers')
plt.title('Total Borrowers By Number of Defaulted Loans',fontweight = 'bold',fontsize = '16')
plt.show()
```



28. Active Loan Overlap. Finds borrowers with more than 1 loan currently in Current status, including total_acc and revol_util. Shows borrowers juggling multiple active obligations and potential overextension risk.

The Active Loan Overlap analysis highlights borrowers juggling multiple current loans, revealing potential overextension risk. Most borrowers hold 2–3 active loans, with a smaller segment managing 4, indicating a group under considerable financial strain. Those with more current loans also tend to have higher total accounts and elevated revolving utilization, suggesting that multiple obligations amplify credit exposure. The bar plots clearly show that borrowers with 4 active loans carry the highest average utilization, pointing to heightened repayment vulnerability. While managing multiple accounts can reflect financial activity, it also signals the need for careful monitoring to prevent defaults. Overall, borrowers with overlapping loans and high utilization form a critical group for risk assessment and targeted interventions.

In [83]:
```
query = """ SELECT t1.borrower_id,ROUND(AVG(t2.total_acc)) AS 'avg_total_acc' ,ROUND(AVg(t1.revol_util),2) AS 'avg_revol_util',
            COUNT(t1.loan_id)  AS 'num_current_loans'
            FROM loans t1
            JOIN borrowers t2
            ON t1.borrower_id = t2.borrower_id
            WHERE t1.loan_status = 'current'
            GROUP BY t1.borrower_id
            HAVING num_current_loans > 1 AND avg_total_acc > 15
            ORDER BY num_current_loans DESC;"""

df = run_query(conn,query)
df
```

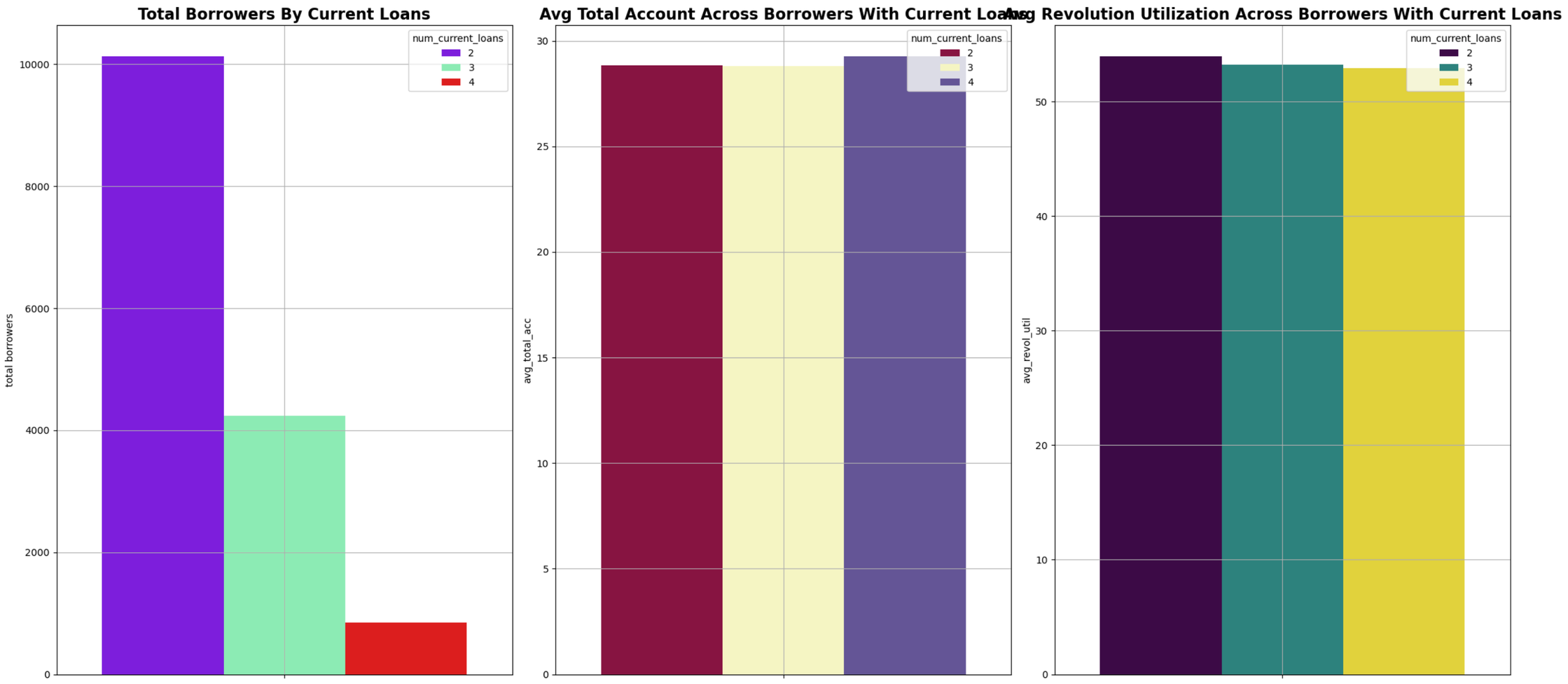
Out[83]:

	borrower_id	avg_total_acc	avg_revol_util	num_current_loans
0	47873	23	32.68	4
1	48036	27	49.70	4
2	48337	21	48.80	4
3	48562	20	60.80	4
4	48845	16	62.18	4
...	...	...	...	...
15219	49904	38	40.55	2
15220	49911	34	64.45	2
15221	49912	25	39.05	2
15222	49921	28	59.20	2
15223	49923	41	39.80	2

15224 rows × 4 columns

In [84]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 3,figsize = (22,10))

sns.barplot(data = df,hue = 'num_current_loans',y = 'borrower_id',estimator = np.count_nonzero,palette = 'rainbow',ax = ax[0])
ax[0].grid(True)
ax[0].set_title('Total Borrowers By Current Loans',fontweight = 'bold',fontsize = '16')
ax[0].set_ylabel('total borrowers')

sns.barplot(data = df,hue = 'num_current_loans',y = 'avg_total_acc',estimator = np.mean,palette = 'Spectral',ax = ax[1],errorbar = None)
ax[1].grid(True)
```

```
ax[1].set_title('Avg Total Account Across Borrowers With Current Loans',fontweight = 'bold',fontsize = '16')

sns.barplot(data = df,hue = 'num_current_loans',y = 'avg_revol_util',estimator = np.mean,palette = 'viridis',ax = ax[2],errorbar = None)
ax[2].grid(True)
ax[2].set_title('Avg Revolution Utilization Across Borrowers With Current Loans',fontweight = 'bold',fontsize = '16')

plt.tight_layout()
plt.show()
```



29. Verification + Income + Defaults for High-Income Borrowers. Compares default rates for borrowers with annual_inc > 100k across verification_status and grade. Evaluates if income verification effectively reduces default risk among wealthy borrowers; helps detect gaps in verification policies.

For high-income borrowers earning over $100k, we observe an interesting pattern when comparing loan defaults across verification status and grade. While verified borrowers generally have slightly higher average defaulted amounts, they also account for a larger number of defaulted borrowers, especially in mid-risk grades like C and D. This suggests that verification alone does not fully protect lenders from defaults among wealthy borrowers—loan grade and risk profile play a stronger role. Pie charts further reveal that verified borrowers make up a bigger share of both defaulted loans and defaulting borrowers, indicating that even financially strong individuals can default if the loan terms are riskier. Overall, the analysis highlights that income verification is necessary but not sufficient, and combining it with careful grading and risk assessment is crucial for managing high-income borrower portfolios effectively.

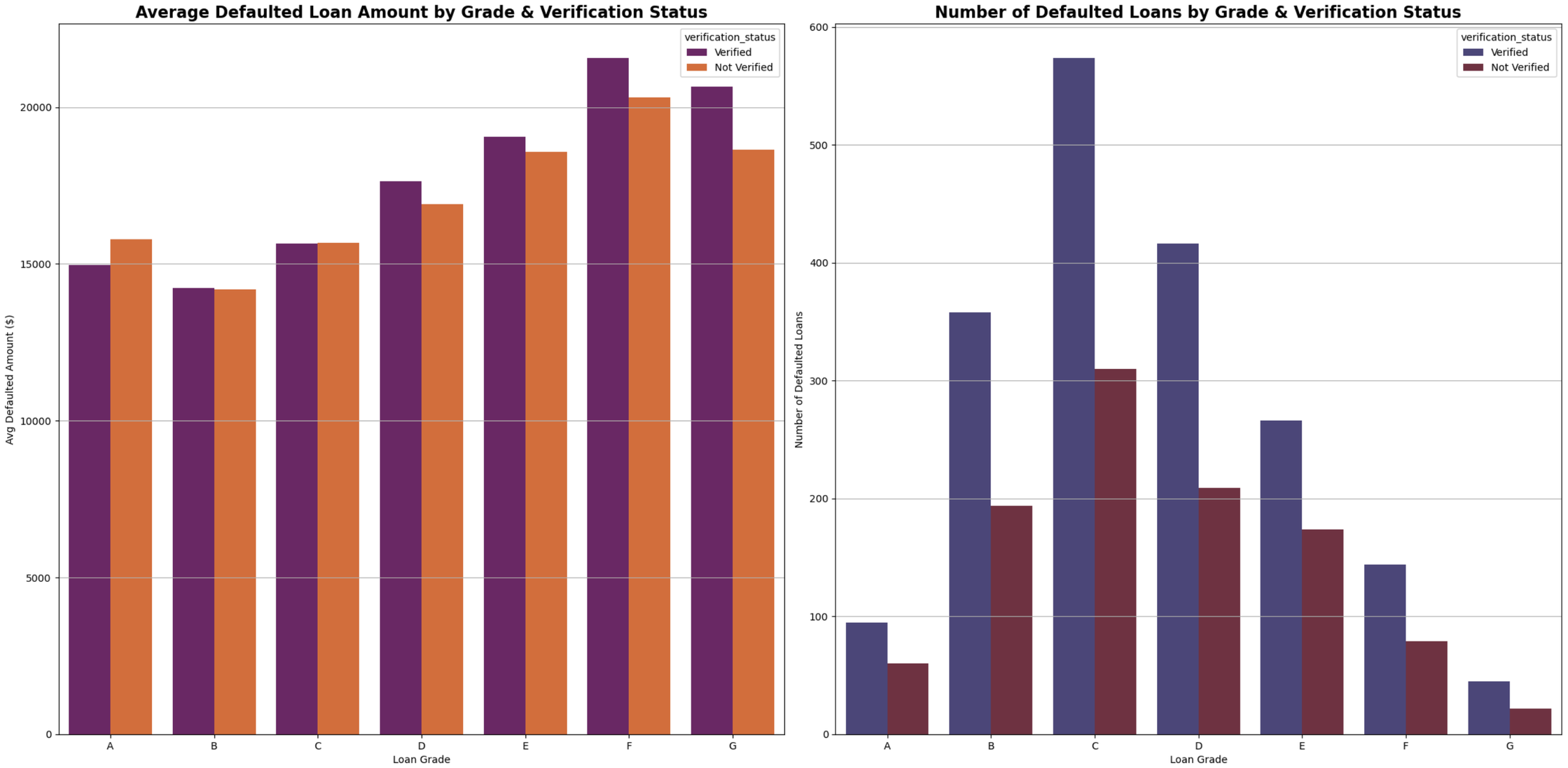
In [85]:
```
query = """ SELECT grade,verification_status,ROUND(AVG(loan_amnt),2) AS 'avg_defaulted_amount',
            COUNT(loan_id) AS 'num_defaulted_loans',
             ROUND(
                   COUNT(loan_id) * 100 /
                   (SELECT COUNT(loan_id)
                    FROM loans t
                    JOIN borrowers b ON t.borrower_id = b.borrower_id
                    WHERE t.loan_status = 'charged off' AND b.annual_inc > 100000)
               ,2) AS percent_default_loans,
            ROUND(
                   COUNT(DISTINCT t1.borrower_id) * 100 /
                   (SELECT COUNT(DISTINCT t.borrower_id)
                    FROM loans t
                    JOIN borrowers b ON t.borrower_id = b.borrower_id
                    WHERE t.loan_status = 'charged off' AND b.annual_inc > 100000)
               ,2) AS percent_default_borrowers
           FROM loans t1
           JOIN borrowers t2
           ON t1.borrower_id = t2.borrower_id
           WHERE t1.loan_status = 'charged off' AND t2.annual_inc > 100000
           GROUP BY t1.grade,t2.verification_status
           ORDER BY grade,verification_status DESC;"""

df = run_query(conn,query)
df
```

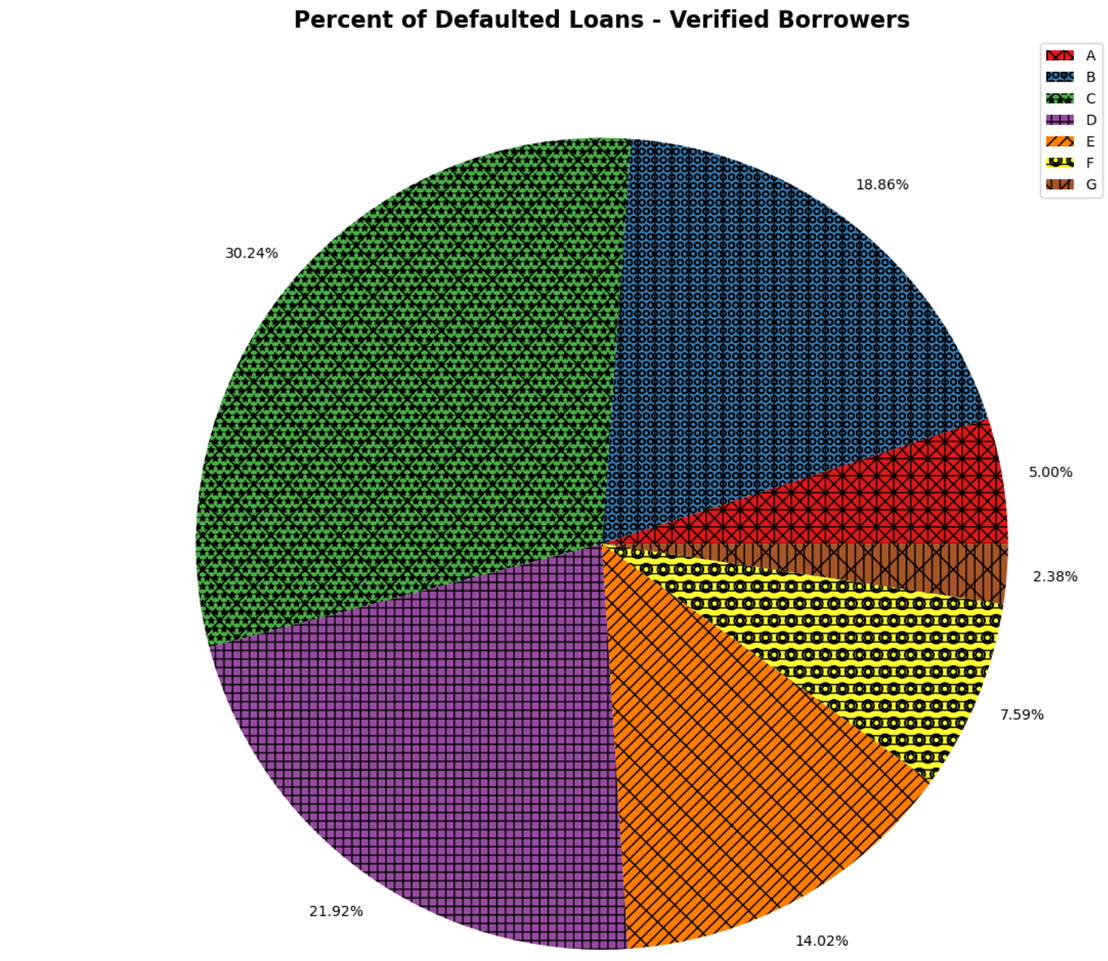
Out[85]:

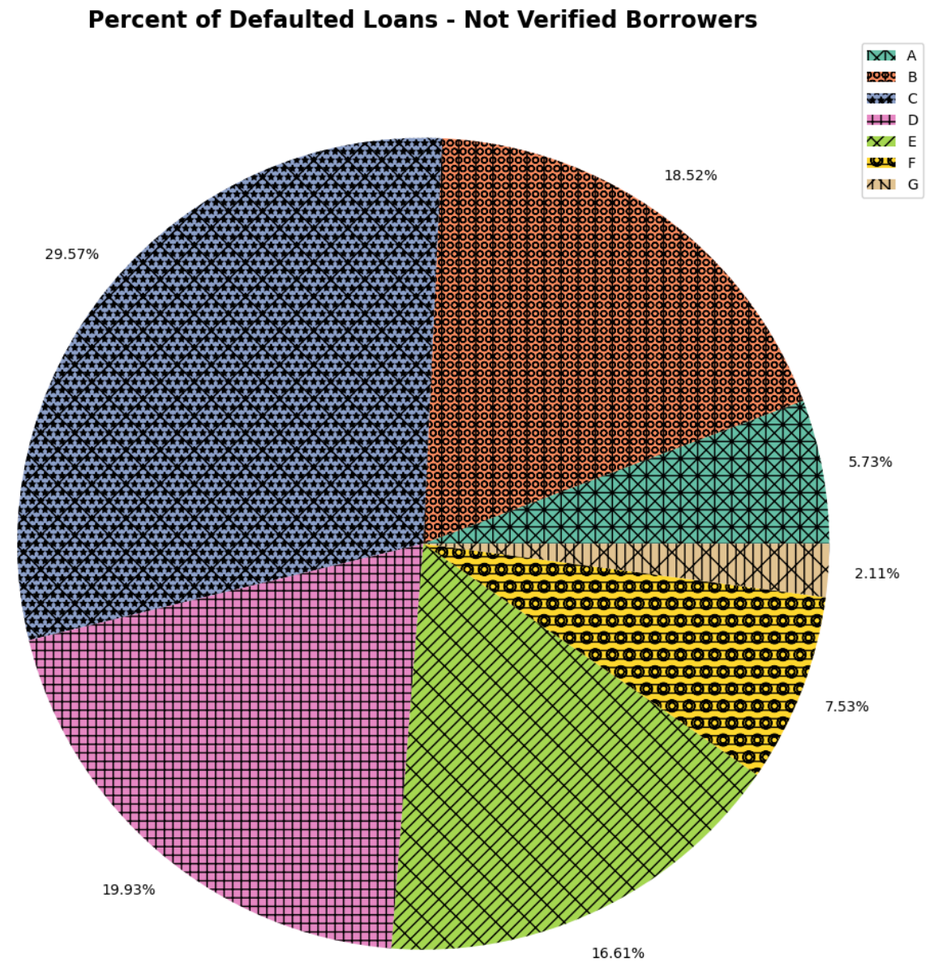
	grade	verification_status	avg_defaulted_amount	num_defaulted_loans	percent_default_loans	percent_default_borrowers
0	A	Verified	14965.53	95	3.22	3.56
1	A	Not Verified	15798.33	60	2.04	2.27
2	B	Verified	14229.40	358	12.15	13.22
3	B	Not Verified	14195.10	194	6.59	7.24
4	C	Verified	15654.44	574	19.48	21.41
5	C	Not Verified	15672.50	310	10.52	11.14
6	D	Verified	17646.81	416	14.12	15.46
7	D	Not Verified	16910.05	209	7.09	7.69
8	E	Verified	19057.33	266	9.03	9.93
9	E	Not Verified	18565.66	174	5.91	6.48
10	F	Verified	21580.73	144	4.89	5.38
11	F	Not Verified	20319.94	79	2.68	2.96
12	G	Verified	20655.00	45	1.53	1.71
13	G	Not Verified	18639.77	22	0.75	0.83

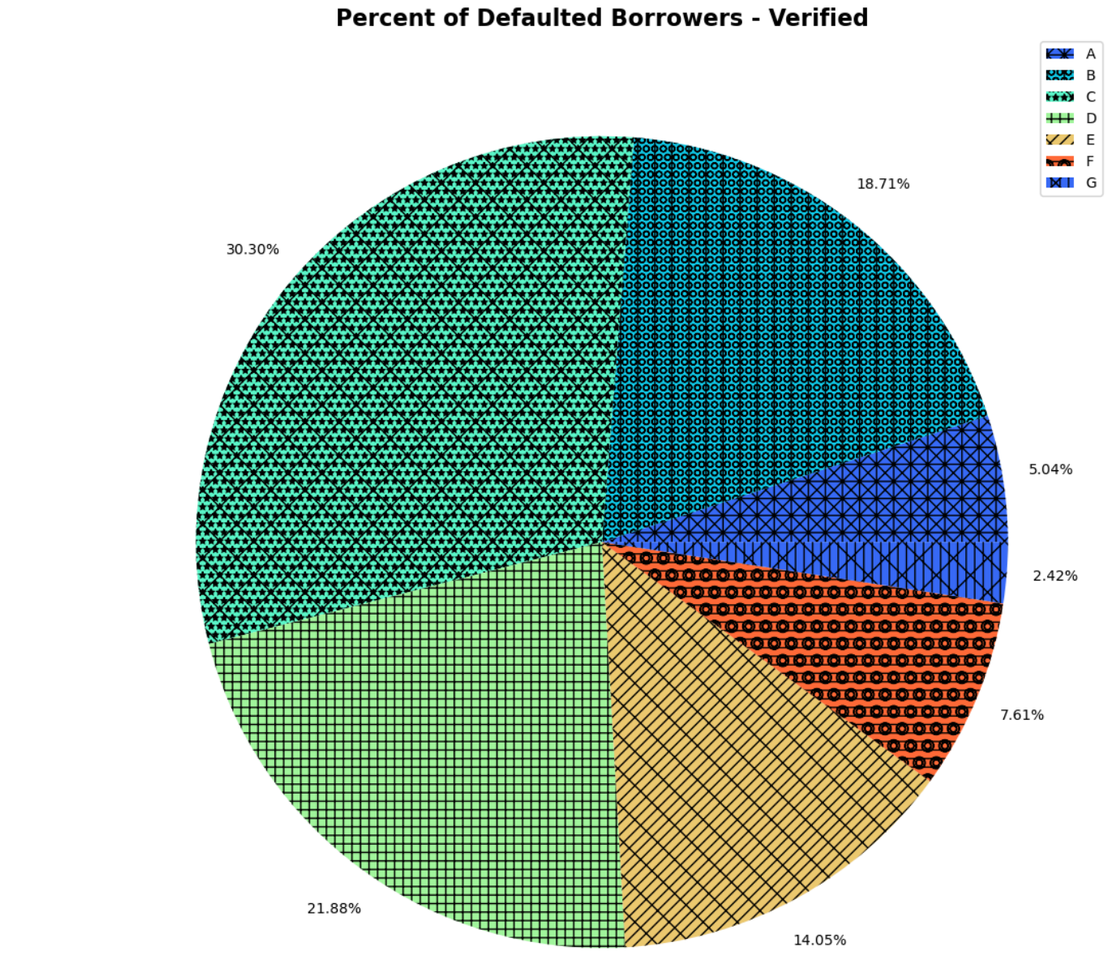
In [86]:
```
fig,ax = plt.subplots(nrows = 3,ncols = 2 ,figsize = (22,32))

sns.barplot(data = df,x = 'grade',y = 'avg_defaulted_amount',hue = 'verification_status',palette = 'inferno',ax = ax[0,0])
ax[0,0].set_title('Average Defaulted Loan Amount by Grade & Verification Status', fontsize = 16, fontweight = 'bold')
ax[0,0].set_xlabel('Loan Grade')
ax[0,0].set_ylabel('Avg Defaulted Amount ($)')
ax[0,0].grid(True, axis = 'y')

sns.barplot(data = df,x = 'grade',y = 'num_defaulted_loans',hue = 'verification_status',palette = 'icefire',ax = ax[0,1])
ax[0,1].set_title('Number of Defaulted Loans by Grade & Verification Status', fontsize = 16, fontweight = 'bold')
ax[0,1].set_xlabel('Loan Grade')
ax[0,1].set_ylabel('Number of Defaulted Loans')
```

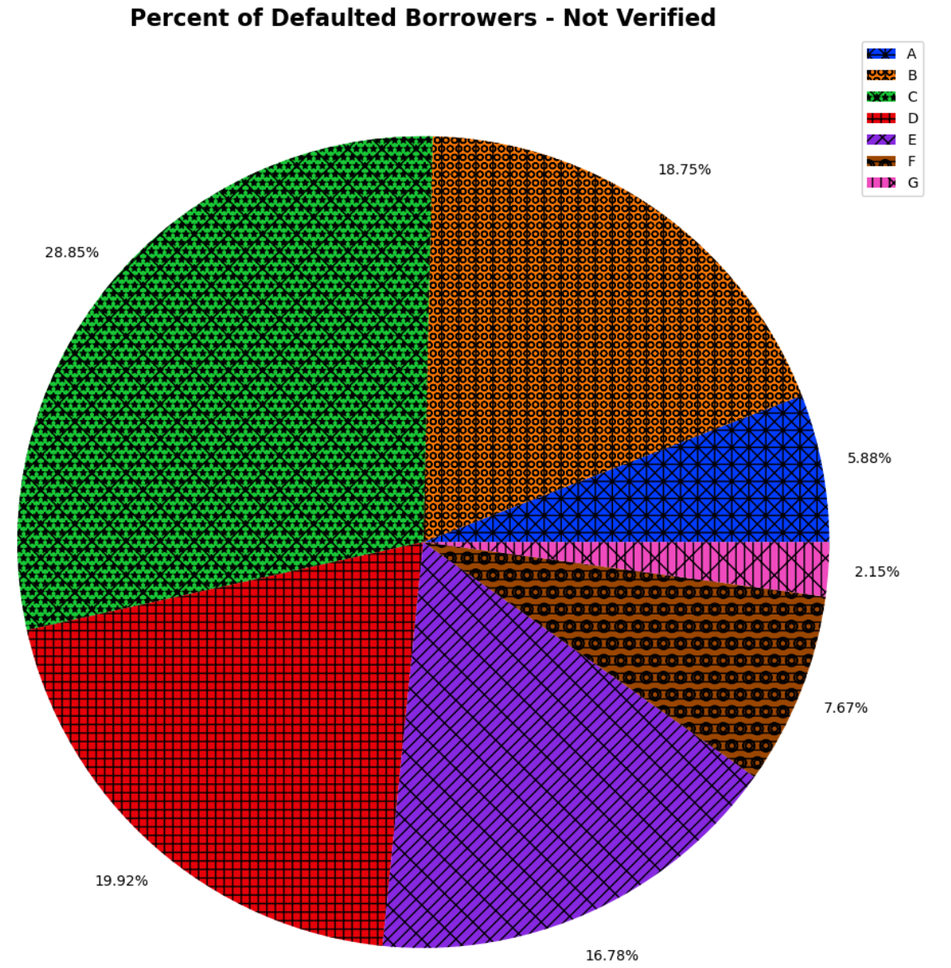
```
ax[0,1].grid(True, axis = 'y')

verified_pie = df[df['verification_status'] == 'Verified']

ax[1,0].pie(verified_pie['percent_default_loans'],autopct = '%1.2f%%',colors = sns.color_palette('Set1'),
            hatch = ['xX+','O|O','*x*','++','/x/','-Oo-','|X|'],pctdistance = 1.12)
ax[1,0].set_title('Percent of Defaulted Loans - Verified Borrowers', fontsize=16, fontweight='bold')
ax[1,0].legend(labels = verified_pie['grade'])


not_verified_pie = df[df['verification_status'] == 'Not Verified']

ax[1,1].pie(not_verified_pie['percent_default_loans'],autopct = '%1.2f%%',colors = sns.color_palette('Set2'),
            hatch = ['xX+','O|O','*x*','++','/x/','-Oo-','|X|'],pctdistance = 1.12)
ax[1,1].legend(labels = verified_pie['grade'])
ax[1,1].set_title('Percent of Defaulted Loans - Not Verified Borrowers', fontsize = 16, fontweight = 'bold')

ax[2,0].pie(verified_pie['percent_default_borrowers'],autopct = '%1.2f%%',colors = sns.color_palette('rainbow'),
            hatch = ['xX+','O|O','*x*','++','/x/','-Oo-','|X|'],pctdistance = 1.12)
ax[2,0].legend(labels = verified_pie['grade'])
ax[2,0].set_title('Percent of Defaulted Borrowers - Verified', fontsize = 16, fontweight = 'bold')

ax[2,1].pie(not_verified_pie['percent_default_borrowers'],autopct = '%1.2f%%',colors = sns.color_palette('bright'),
            hatch = ['xX+','O|O','*x*','++','/x/','-Oo-','|X|'],pctdistance = 1.12)
ax[2,1].legend(labels = verified_pie['grade'])
ax[2,1].set_title('Percent of Defaulted Borrowers - Not Verified', fontsize = 16, fontweight = 'bold')

plt.tight_layout()
plt.show()
```











30. Repeat Default Trend Analysis. Calculates the difference in loan_amnt between a borrower’s current defaulted loan and previous loan. Shows if repeat defaulters are taking larger loans over time, indicating rising financial stress.

`The repeat default analysis reveals that borrower's loan behaviors vary significantly over time. A large portion of repeat defaulters are taking bigger loans than their previous ones, indicating rising financial stress or over-leveraging, while others are borrowing smaller amounts, possibly reflecting tighter borrowing limits or cautious behavior. Boxplots show that the magnitude of increase in loan amounts is often substantial, with some borrowers escalating by over $15,000, whereas decreases can be equally dramatic. Scatterplots highlight that higher previous loans don't always prevent larger subsequent defaults, suggesting that repeat defaulters may not learn from past risk. On average, borrowers who increase their loan size show a positive upward trend in default amounts, underlining the need for lenders to monitor borrowing patterns over time. Overall, this analysis emphasizes that tracking loan size changes among repeat defaulters can provide early warning signals of financial stress and potential risk escalation.

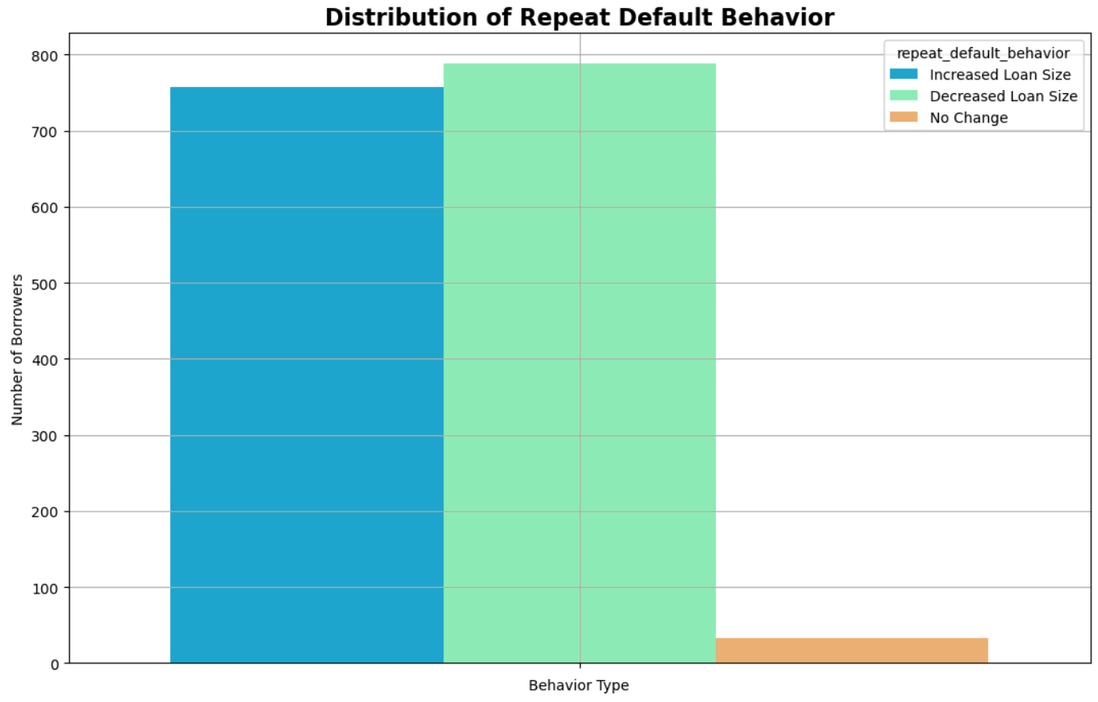
In [87]:
```
query = """  WITH defaulted_loans AS (SELECT borrower_id,loan_id,loan_amnt,
             LAG(loan_amnt) OVER (PARTITION BY borrower_id ORDER BY loan_id) AS prev_loan_amnt
             FROM loans
             WHERE loan_status = 'charged off')


             SELECT borrower_id,loan_id,loan_amnt AS current_default_loan_amnt,
                   prev_loan_amnt,(loan_amnt - prev_loan_amnt) AS loan_amnt_diff,
                   CASE
                      WHEN loan_amnt - prev_loan_amnt > 0 THEN 'Increased Loan Size'
                      WHEN loan_amnt - prev_loan_amnt < 0 THEN 'Decreased Loan Size'
                      WHEN loan_amnt - prev_loan_amnt = 0 THEN 'No Change'
                   END AS 'repeat_default_behavior'
             FROM defaulted_loans
             WHERE prev_loan_amnt IS NOT NULL
             ORDER BY borrower_id;"""

df = run_query(conn,query)
df
```

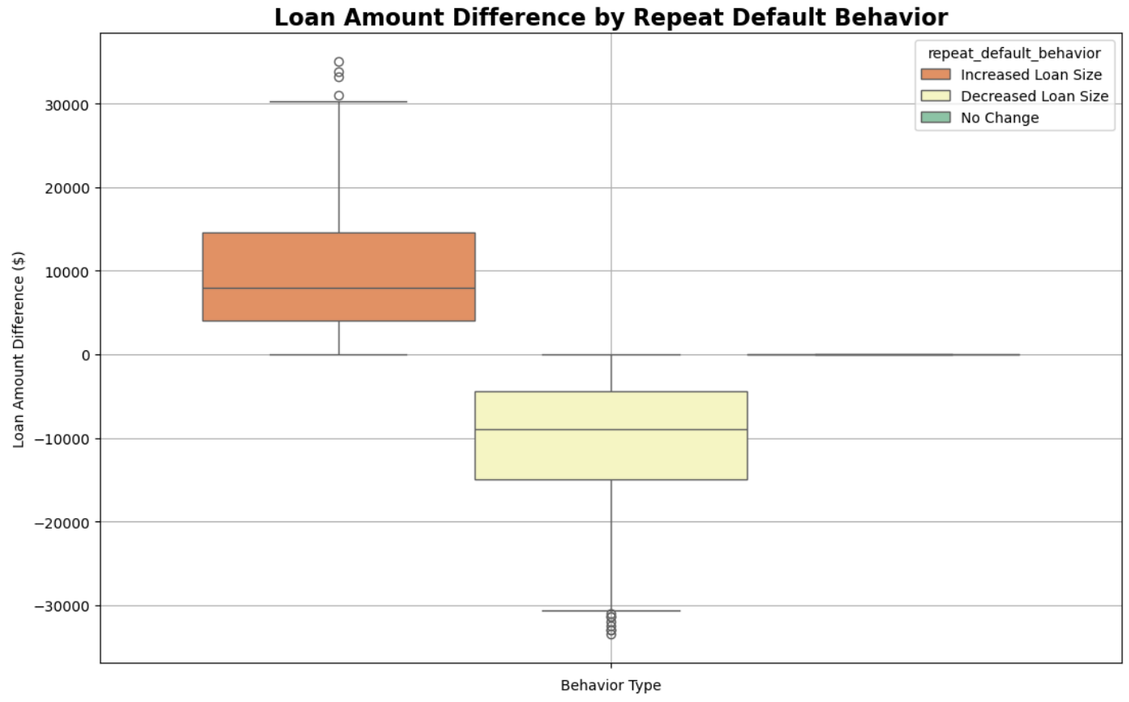
Out[87]:

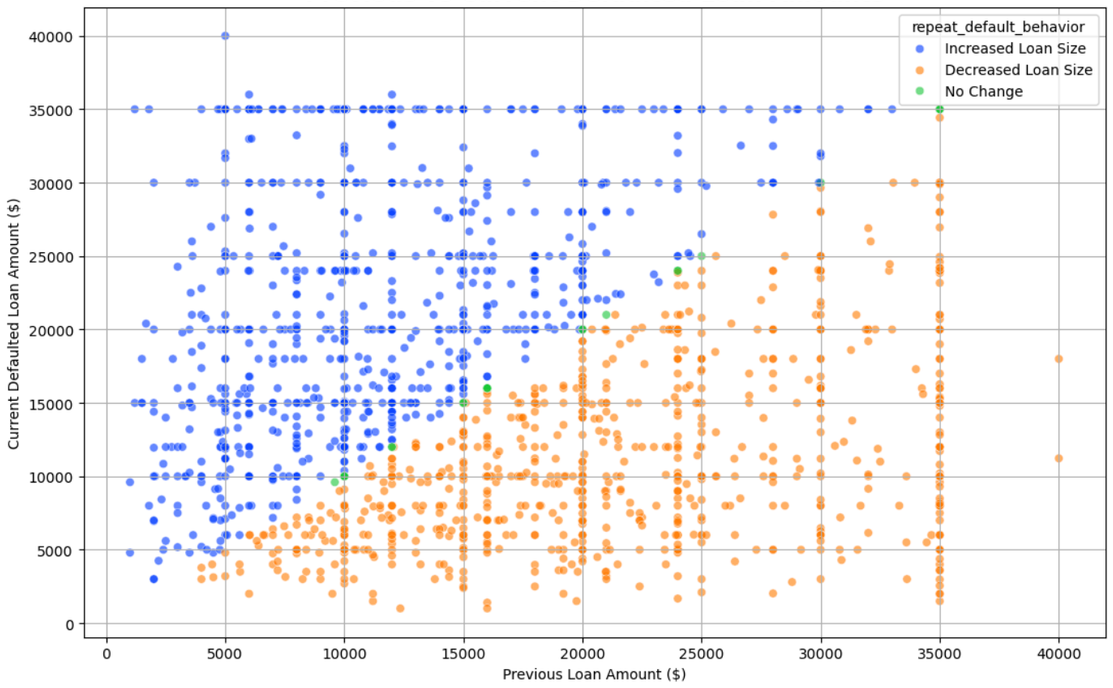
	borrower_id	loan_id	current_default_loan_amnt	prev_loan_amnt	loan_amnt_diff	repeat_default_behavior
0	2	LN000007	23350	8000	15350	Increased Loan Size
1	4	LN000011	3000	15000	-12000	Decreased Loan Size
2	20	LN000057	25000	15400	9600	Increased Loan Size
3	30	LN000083	13000	4800	8200	Increased Loan Size
4	57	LN000157	12000	26700	-14700	Decreased Loan Size
...	...	...	...	...	...	...
1574	49772	LN124496	7475	20000	-12525	Decreased Loan Size
1575	49772	LN124497	6600	7475	-875	Decreased Loan Size
1576	49888	LN124784	15000	8425	6575	Increased Loan Size
1577	49898	LN124813	29900	13075	16825	Increased Loan Size
1578	49944	LN124927	10000	9000	1000	Increased Loan Size

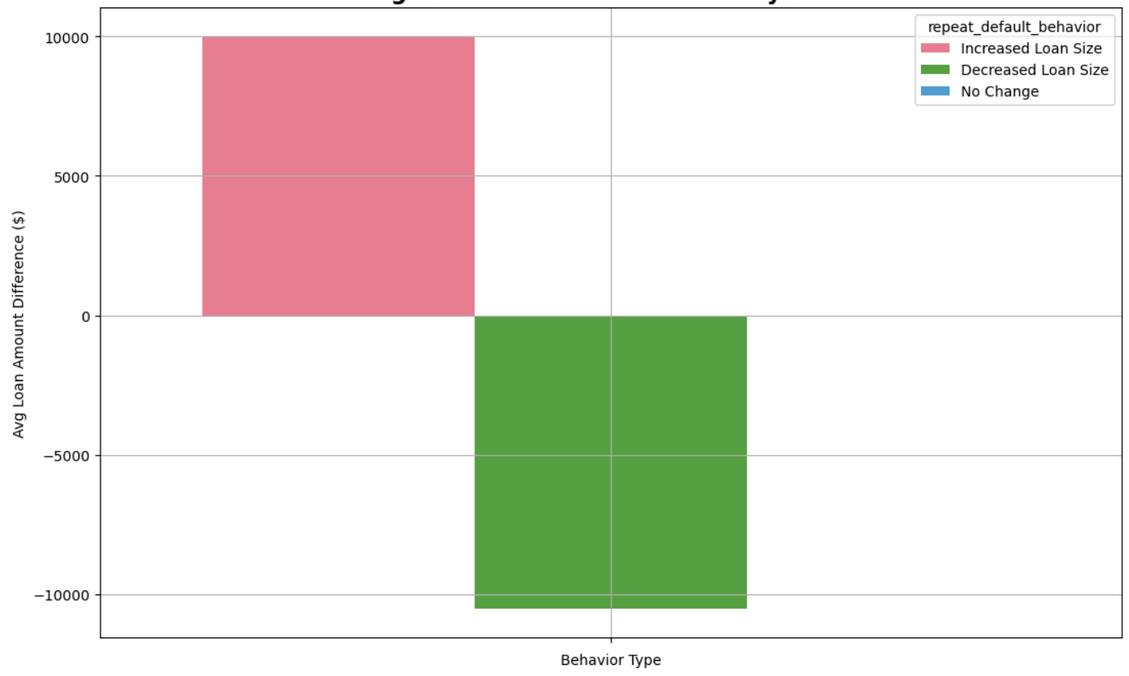
1579 rows × 6 columns

In [88]:
```
fig,ax = plt.subplots(nrows = 2,ncols = 2,figsize = (22,14))

sns.barplot(data = df,hue = "repeat_default_behavior",y = 'borrower_id',estimator = np.count_nonzero,palette = "rainbow",ax = ax[0,0])
ax[0,0].set_title("Distribution of Repeat Default Behavior",fontsize = '16',fontweight = 'bold')
ax[0,0].grid(True)
ax[0,0].set_xlabel("Behavior Type")
ax[0,0].set_ylabel("Number of Borrowers")

sns.boxplot(data = df, hue = "repeat_default_behavior", y = "loan_amnt_diff", palette = "Spectral",ax = ax[0,1])
ax[0,1].set_title("Loan Amount Difference by Repeat Default Behavior",fontweight = 'bold',fontsize = '16')
ax[0,1].grid(True)
ax[0,1].set_xlabel("Behavior Type")
ax[0,1].set_ylabel("Loan Amount Difference ($)")

sns.scatterplot(data = df, x = "prev_loan_amnt", y = "current_default_loan_amnt",hue = "repeat_default_behavior",
                ax = ax[1,0],alpha = 0.6, palette = "bright")
ax[1,0].set_title("Previous vs Current Defaulted Loan Amounts",fontweight = 'bold',fontsize = '16')
ax[1,0].grid(True)
ax[1,0].set_xlabel("Previous Loan Amount ($)")
ax[1,0].set_ylabel("Current Defaulted Loan Amount ($)")

sns.barplot(data = df, hue = "repeat_default_behavior", y = "loan_amnt_diff", estimator = np.mean, palette ="husl",ax = ax[1,1],
           errorbar = None)
ax[1,1].set_title("Average Loan Amount Difference by Behavior",fontweight = 'bold',fontsize = '16')
ax[1,1].grid(True)
ax[1,1].set_xlabel("Behavior Type")
ax[1,1].set_ylabel("Avg Loan Amount Difference ($)")

plt.tight_layout()
plt.show()
```









31. First Loan Performance vs Current Status. Evaluates the impact of first loan performance on subsequent loan outcomes to identify trends in repeat defaults and repayment behavior; informs risk models and borrower scoring.

The analysis shows that a borrower’s first loan performance strongly influences subsequent loan outcomes. Borrowers whose first loan was charged off have a high chance of repeating defaults, with 46.9% of their next loans also being charged off, while only 18.8% manage to fully repay. Conversely, those whose first loan was fully paid or current are far more likely to continue good repayment behavior, with over 58% of fully paid borrowers staying fully paid and 72% of current borrowers remaining current on their next loan. Late or in-grace-period first loans show mixed outcomes, highlighting moderate risk for future defaults. The heatmap and stacked bar chart reveal clear trends: early loan behavior is a strong predictor for repeat default risk and can inform risk modeling, credit scoring, and targeted monitoring strategies.

In [89]:
```
query = """ WITH loan_history AS (SELECT borrower_id,loan_id,loan_status,
            FIRST_VALUE(loan_status) OVER (PARTITION BY borrower_id ORDER BY loan_id) AS first_loan_status
            FROM loans)


            SELECT first_loan_status,loan_status AS subsequent_loan_status,COUNT(loan_id) AS num_loans,
                   ROUND(100 * COUNT(loan_id) / SUM(COUNT(loan_id)) OVER(PARTITION BY first_loan_status),2)
                   AS 'percent_within_group'
            FROM loan_history
            GROUP BY first_loan_status, loan_status
            ORDER BY first_loan_status, percent_within_group DESC;"""

df = run_query(conn,query)
df["percent_within_group"] = df["percent_within_group"].astype('float')
df
```

Out[89]:

	first_loan_status	subsequent_loan_status	num_loans	percent_within_group
0	charged off	charged off	6729	46.90
1	charged off	current	4657	32.46
2	charged off	fully paid	2696	18.79
3	charged off	late	196	1.37
4	charged off	in grace period	69	0.48
5	current	current	49011	72.48
6	current	fully paid	12927	19.12
7	current	charged off	4516	6.68
8	current	late	834	1.23
9	current	in grace period	329	0.49
10	fully paid	fully paid	23083	58.81
11	fully paid	current	12725	32.42
12	fully paid	charged off	2796	7.12
13	fully paid	late	445	1.13
14	fully paid	in grace period	200	0.51
15	in grace period	in grace period	436	39.14
16	in grace period	current	351	31.51
17	in grace period	fully paid	225	20.20
18	in grace period	charged off	86	7.72
19	in grace period	late	16	1.44
20	late	late	1063	41.72
21	late	current	833	32.69
22	late	fully paid	459	18.01
23	late	charged off	181	7.10
24	late	in grace period	12	0.47

In [90]:
```
pivot_df1 = df.pivot_table(columns = "first_loan_status",index = "subsequent_loan_status",values = "percent_within_group")
pivot_df1
```

Out[90]:

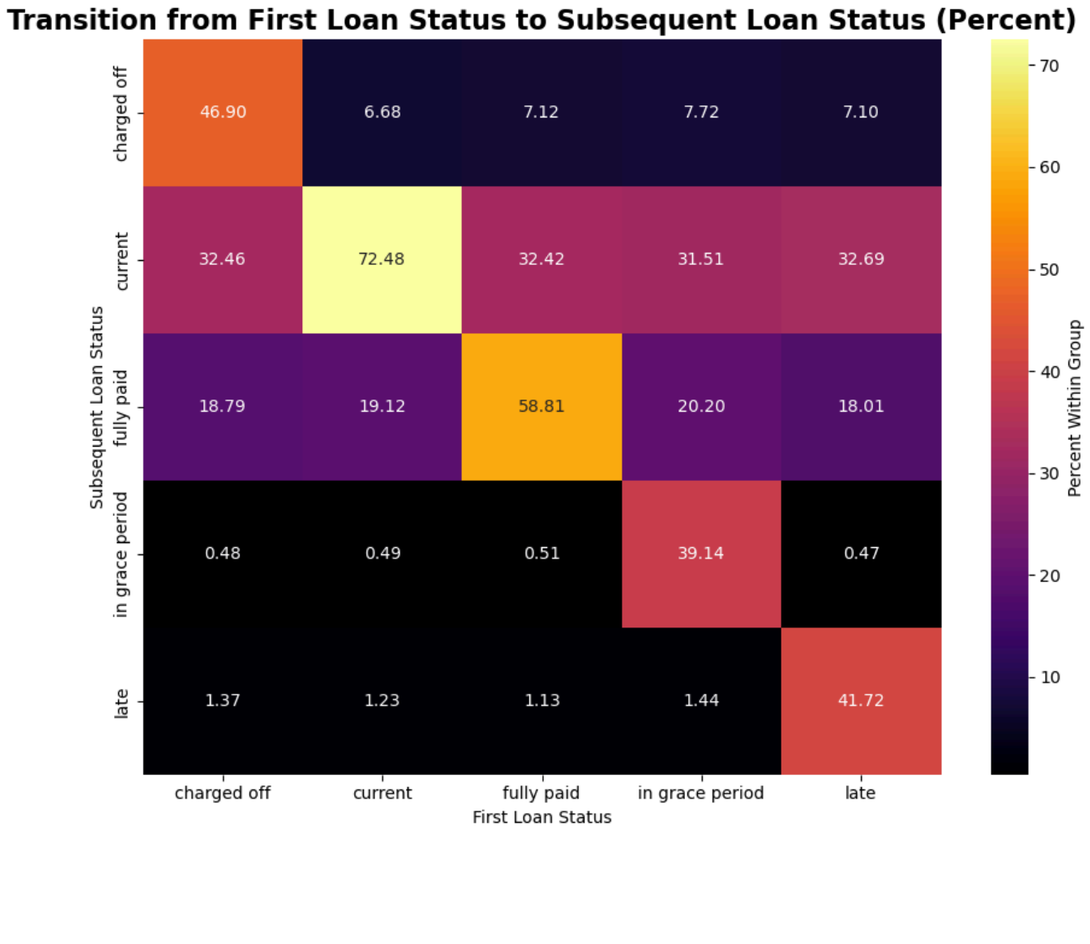
first_loan_status	charged off	current	fully paid	in grace period	late
subsequent_loan_status					
charged off	46.90	6.68	7.12	7.72	7.10
current	32.46	72.48	32.42	31.51	32.69
fully paid	18.79	19.12	58.81	20.20	18.01
in grace period	0.48	0.49	0.51	39.14	0.47
late	1.37	1.23	1.13	1.44	41.72

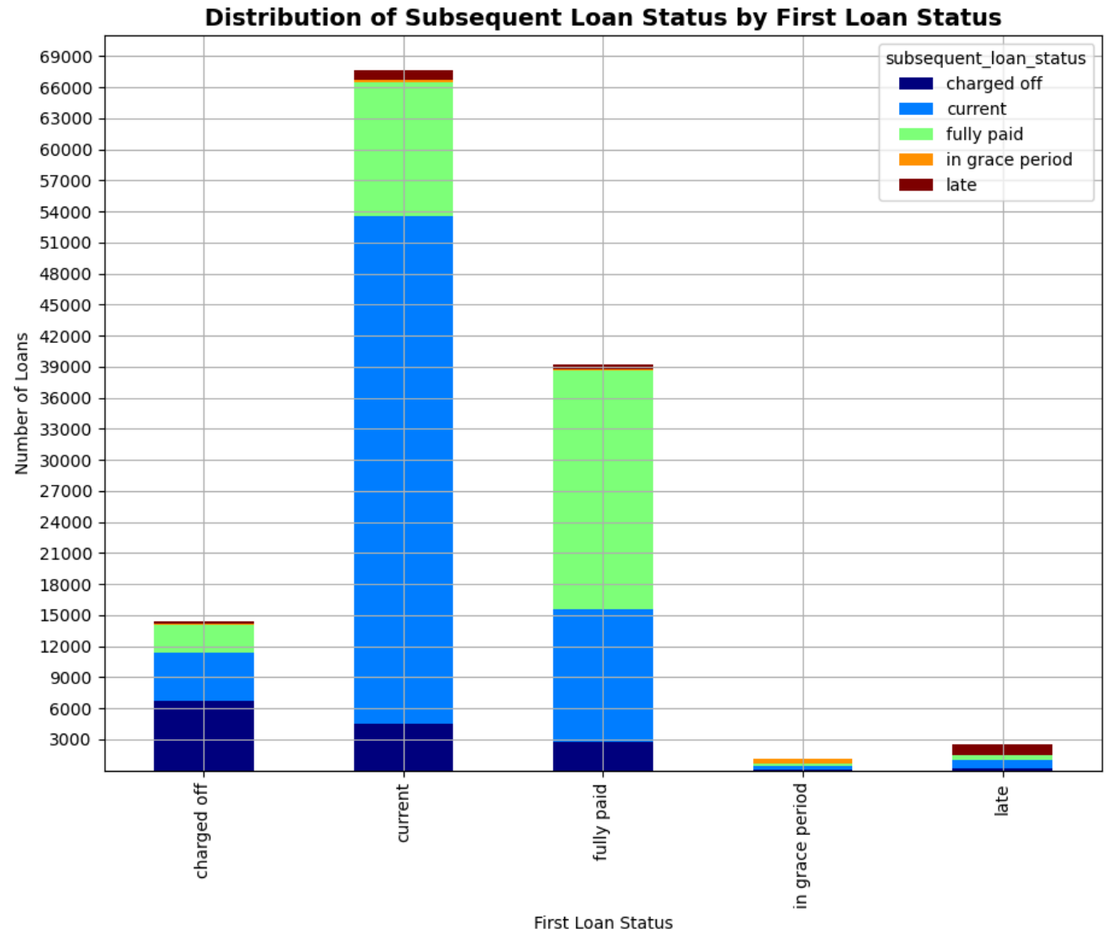
In [91]:
```
pivot_df2 = df.pivot_table(index = "first_loan_status",columns = "subsequent_loan_status",values = "num_loans")
pivot_df2
```

Out[91]:

subsequent_loan_status	charged off	current	fully paid	in grace period	late
first_loan_status					
charged off	6729.0	4657.0	2696.0	69.0	196.0
current	4516.0	49011.0	12927.0	329.0	834.0
fully paid	2796.0	12725.0	23083.0	200.0	445.0
in grace period	86.0	351.0	225.0	436.0	16.0
late	181.0	833.0	459.0	12.0	1063.0

In [92]:
```
fig,ax = plt.subplots(nrows = 1,ncols = 2,figsize = (18,8))

sns.heatmap(pivot_df1, annot = True, fmt = "1.2f", cmap = "inferno", cbar_kws = {'label': 'Percent Within Group'},ax = ax[0])
ax[0].set_title("Transition from First Loan Status to Subsequent Loan Status (Percent)", fontsize = 16, fontweight = "bold")
ax[0].set_xlabel("First Loan Status")
ax[0].set_ylabel("Subsequent Loan Status")

pivot_df2.plot(kind = "bar",stacked = True,colormap = "jet",ax = ax[1])
ax[1].set_title("Distribution of Subsequent Loan Status by First Loan Status", fontsize = 14, fontweight = "bold")
ax[1].set_xlabel("First Loan Status")
ax[1].set_ylabel("Number of Loans")
ax[1].set_yticks(ticks = range(3000,70000,3000))
ax[1].grid(True)

plt.tight_layout()
plt.show()
```





32. Defaulters vs Paid. Tests if income,interest rate,dti truly ensures repayment or if defaults occur across levels.

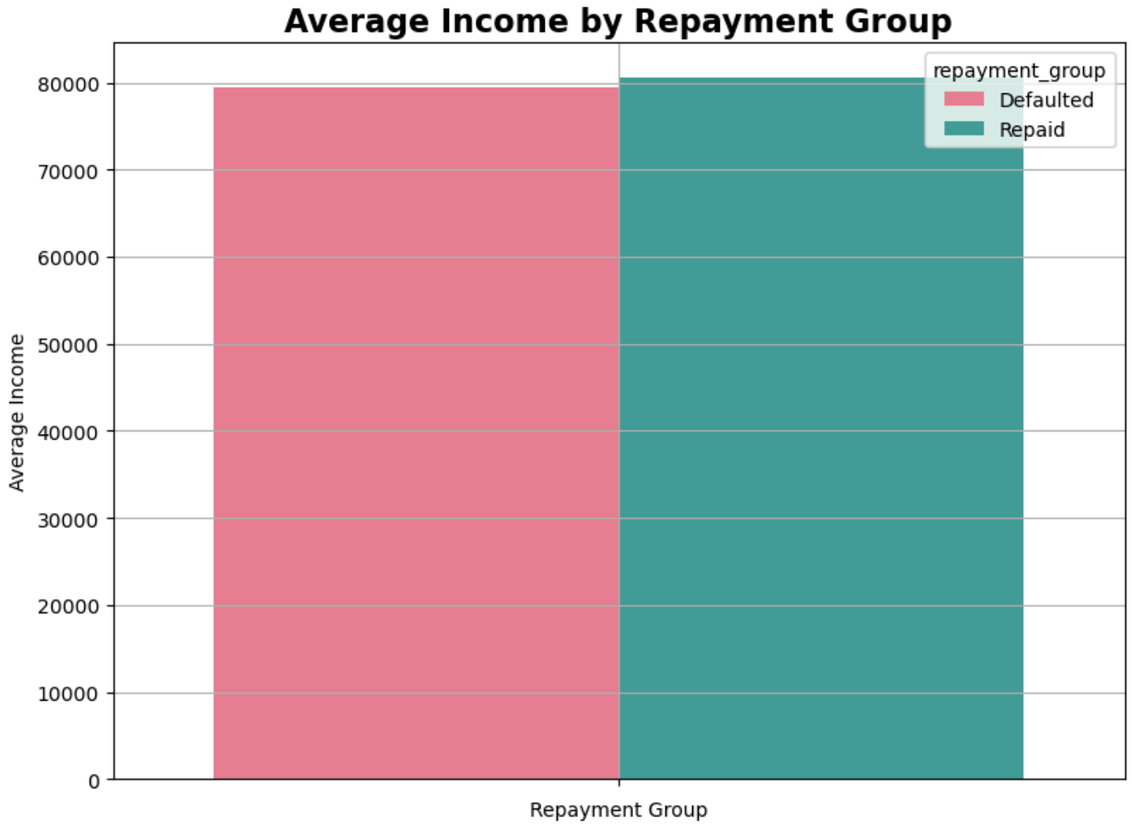
The analysis shows that while income, interest rate, and debt-to-income (DTI) influence repayment behavior, they do not fully guarantee it. Borrowers who defaulted had slightly lower average incomes ($79,360) than those who repaid ($80,540), but faced higher interest rates (15.7% vs 12.1%) and higher DTI (21.7% vs 18.5%). This indicates that elevated borrowing costs and financial strain increase default risk, even among relatively well-off borrowers. Despite these differences, a significant portion (36.2%) of borrowers still defaulted, highlighting that income alone is not enough to predict repayment. The pie chart confirms that most borrowers (82.2%) successfully repay, but monitoring interest and DTI remains crucial for identifying potential risks and guiding lending decisions.

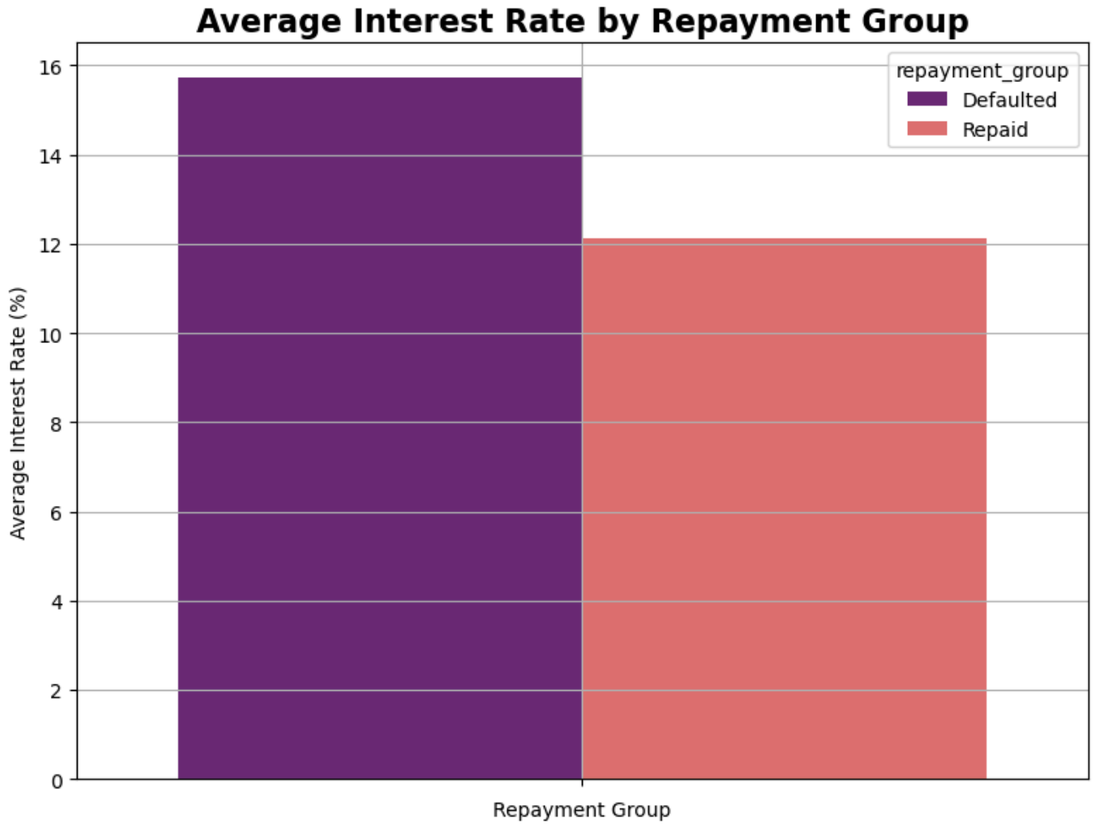
In [93]:
```sql
query = """ SELECT
                CASE
                    WHEN loan_status = 'charged off' THEN 'Defaulted'
                    WHEN loan_status = 'fully paid'  THEN 'Repaid'
                END AS 'repayment_group',
                ROUND(AVG(annual_inc), 2) AS 'avg_income',
                ROUND(AVG(int_rate), 2) AS 'avg_int_rate',
                ROUND(AVG(dti), 2) AS 'avg_dti',
                ROUND( COUNT(DISTINCT(t1.borrower_id)) * 100 /
                       (SELECT COUNT(DISTINCT(borrower_id)) FROM loans WHERE loan_status IN ('charged off', 'fully paid'))
                     ,2)
                AS 'percent_borrowers'
            FROM loans t1
            JOIN borrowers t2
            ON t1.borrower_id = t2.borrower_id
            WHERE loan_status IN ('charged off', 'fully paid')
            GROUP BY repayment_group
            ORDER BY repayment_group;"""

df = run_query(conn,query)
df
```

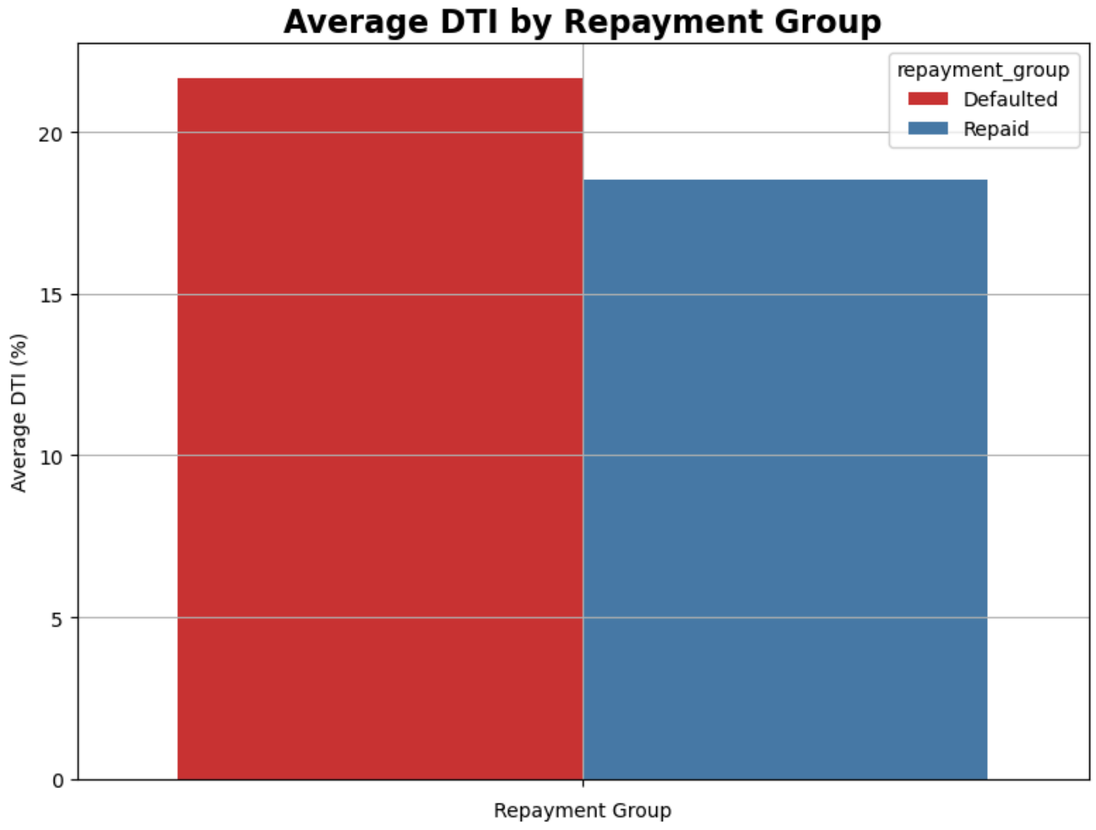
Out[93]:

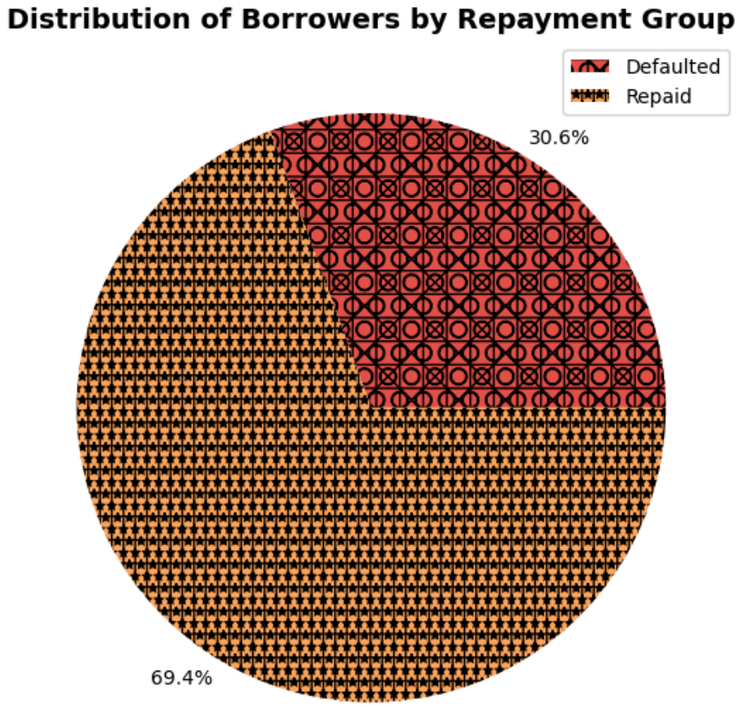
	repayment_group	avg_income	avg_int_rate	avg_dti	percent_borrowers
0	Defaulted	79359.94	15.72	21.68	36.20
1	Repaid	80539.92	12.11	18.51	82.24

In [94]:
```python
fig, ax = plt.subplots(nrows = 2,ncols = 2, figsize=(16, 12))

sns.barplot(data = df, hue = "repayment_group", y = "avg_income", palette = "husl", ax = ax[0,0])
ax[0,0].set_title("Average Income by Repayment Group", fontsize = 16, fontweight = "bold")
ax[0,0].set_xlabel("Repayment Group")
ax[0,0].set_ylabel("Average Income")
ax[0,0].grid(True)

sns.barplot(data = df, hue = "repayment_group", y = "avg_int_rate", palette = "magma", ax = ax[0,1])
ax[0,1].set_title("Average Interest Rate by Repayment Group", fontsize = 16, fontweight = "bold")
ax[0,1].set_xlabel("Repayment Group")
ax[0,1].set_ylabel("Average Interest Rate (%)")
ax[0,1].grid(True)

sns.barplot(data = df, hue = "repayment_group", y = "avg_dti", palette = "Set1", ax = ax[1,0])
ax[1,0].set_title("Average DTI by Repayment Group", fontsize = 16, fontweight = "bold")
ax[1,0].set_xlabel("Repayment Group")
ax[1,0].set_ylabel("Average DTI (%)")
ax[1,0].grid(True)

ax[1,1].pie(df["percent_borrowers"],autopct = '%1.1f%%',colors = sns.color_palette("Spectral"),
            hatch = ['O+X','*||*'],pctdistance = 1.12)
ax[1,1].legend(labels = df["repayment_group"])
ax[1,1].set_title("Distribution of Borrowers by Repayment Group", fontsize=14, fontweight="bold")

plt.tight_layout()
plt.show()
```









In [ ]:

In [ ]:

E. Advanced Insights & Portfolio Strategy : What strategic insights can improve portfolio health?

Finally, we combine borrower and loan dimensions to uncover hidden patterns. This helps us segment risky borrowers, identify safe bets, and suggest strategies for a healthier loan portfolio.

Summary-

In Area E, we combined borrower and loan factors to uncover patterns that help improve portfolio health and reduce risk. Verification status matters less than loan grade—high-grade loans remain low-risk even if not verified, while low-grade loans are risky regardless. Borrowers with high debt-to-income ratios and high revolving utilization face much higher default risk, and short employment tenure slightly increases risk. Home ownership slightly influences rates and loan amounts, but the biggest risk comes from lower grades rather than ownership type.

Portfolio concentration shows that debt consolidation loans in grades B and C dominate exposure, making these segments critical for monitoring. Multi-grade borrowers (who take loans across grades) show slightly higher defaults and interest rates, indicating potential repeat borrowing and risk. Delinquency history is a strong early warning signal, as past delinquencies significantly increase default likelihood across grades. Finally, the safest lending zones combine high grades with popular purposes like debt consolidation, moving, and home improvement, offering high repayment percentages while maintaining portfolio scale. Overall, these insights guide lenders to focus on stable, high-grade borrowers and carefully monitor risky segments to improve portfolio performance.

In [ ]:

33. Verification × Grade: Which combos default the most? Verifies if income verification meaningfully reduces risk within each grade.

The analysis shows that loan grade is the strongest driver of defaults, with lower grades (F–G) experiencing the highest default percentages, while high grades (A–B) maintain very low defaults. Income verification has only a small effect—verified and non-verified borrowers in the same grade show similar default rates, indicating that verification alone does not significantly reduce risk. Mid-grades like C–D see noticeable jumps in defaults, suggesting these borrowers carry moderate risk that requires careful monitoring. Interestingly, average interest rates rise steadily as grade decreases, correlating with higher defaults and reflecting lenders' attempt to price for risk. Scatterplots highlight this relationship, showing that loans with higher interest rates consistently have higher default percentages regardless of verification. Overall, the results suggest that while verification provides some assurance, loan grade and interest rate are far better indicators of default risk, guiding lenders to focus on loan quality and borrower risk profile.

In [95]:
```
query = """ WITH base AS (
              SELECT l.grade,
                     b.verification_status,
                     l.int_rate,
                     l.loan_status
              FROM loans l
              JOIN borrowers b ON l.borrower_id = b.borrower_id
            )

            SELECT
              grade,
              verification_status,
              COUNT(*) AS 'loan_count',
              ROUND(AVG(CASE WHEN loan_status='charged off' THEN 1 ELSE 0 END)*100,2) AS 'default_pct',
```

```
            ROUND(AVG(int_rate), 2) AS 'avg_int_rate'
          FROM base
          GROUP BY grade, verification_status
          HAVING loan_count >= 100
          ORDER BY grade,verification_status DESC;"""

df = run_query(conn,query)
df
```

Out[95]:

	grade	verification_status	loan_count	default_pct	avg_int_rate
0	A	Verified	15872	2.89	6.68
1	A	Not Verified	8853	2.85	6.69
2	B	Verified	24019	7.07	9.97
3	B	Not Verified	13579	7.15	9.97
4	C	Verified	21854	12.73	13.46
5	C	Not Verified	12493	12.90	13.48
6	D	Verified	9710	19.13	17.61
7	D	Not Verified	5569	19.95	17.59
8	E	Verified	5687	24.51	20.74
9	E	Not Verified	3198	25.48	20.73
10	F	Verified	2024	32.76	24.28
11	F	Not Verified	1196	31.10	24.31
12	G	Verified	550	38.73	27.96
13	G	Not Verified	271	39.85	28.07

In [96]:

```
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(18,12))

sns.barplot(data = df,x = 'grade',y = 'default_pct',hue = 'verification_status',palette = 'rainbow',ax = ax[0,0])
ax[0,0].set_title('Default Percentage by Grade & Verification Status', fontsize = 14, fontweight = 'bold')
ax[0,0].set_ylabel('% of Loans Defaulted')
ax[0,0].set_xlabel('Loan Grade')
ax[0,0].grid(True, axis = 'y')

sns.barplot(data = df,x = 'grade',y = 'loan_count',hue = 'verification_status',palette = 'inferno',ax = ax[0,1])
ax[0,1].set_title('Number of Loans by Grade & Verification Status', fontsize = 14, fontweight = 'bold')
ax[0,1].set_ylabel('Loan Count')
ax[0,1].set_xlabel('Loan Grade')
ax[0,1].grid(True, axis = 'y')

sns.barplot(data = df,x = 'grade',y = 'avg_int_rate',hue = 'verification_status',palette = 'Set1', ax = ax[1,0])
ax[1,0].set_title('Average Interest Rate by Grade & Verification Status', fontsize = 14, fontweight = 'bold')
ax[1,0].set_ylabel('Avg Interest Rate (%)')
ax[1,0].set_xlabel('Loan Grade')
ax[1,0].grid(True, axis = 'y')

sns.scatterplot(data = df,x = 'avg_int_rate',y = 'default_pct',hue = 'verification_status',style = 'grade',
                s = 200,palette = 'bright',ax = ax[1,1])
ax[1,1].set_title('Default % vs Avg Interest Rate', fontsize=14, fontweight='bold')
ax[1,1].set_xlabel('Average Interest Rate (%)')
ax[1,1].set_ylabel('Default %')
ax[1,1].grid(True)

plt.tight_layout()
plt.show()
```

34. DTI × Revolving Utilization Risk Grid. Interaction of leverage (DTI) and utilization often explains defaults better than either alone.

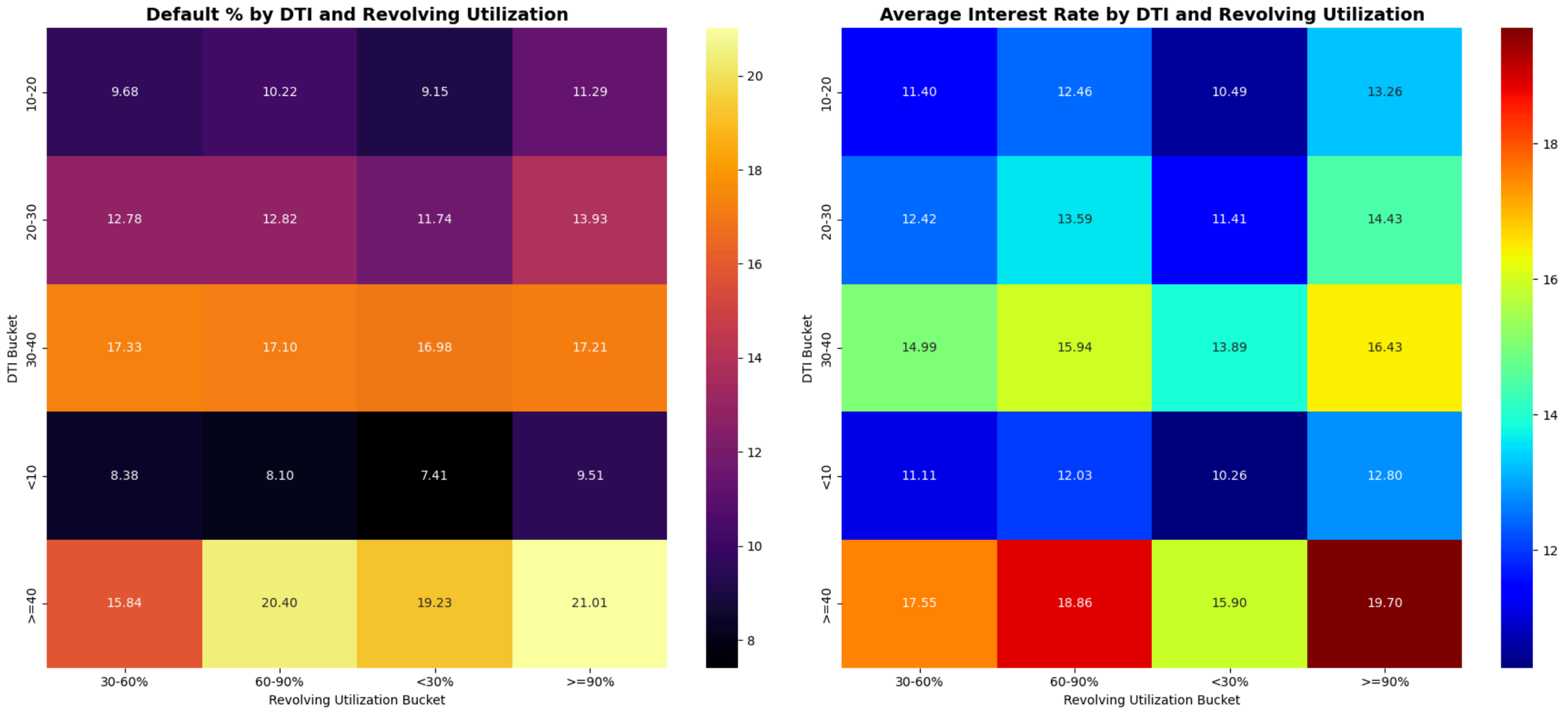
The analysis shows that defaults are strongly influenced by the combination of debt-to-income (DTI) and revolving credit utilization, with high DTI (≥40%) and high utilization (≥90%) borrowers facing the highest default rates above 20%. Lower DTI borrowers (<10%) generally have low defaults, but rising utilization still increases risk, highlighting that over-leveraging matters even for financially stable borrowers. Mid-range DTI and utilization buckets (20–40% DTI, 60–90% utilization) show moderate defaults, suggesting that both metrics together provide a clearer picture of financial stress than either alone. Heatmaps reveal that average interest rates increase steadily with higher DTI and utilization, reflecting lenders risk-based pricing. Overall, the interaction of DTI and revolving utilization effectively identifies borrowers at higher risk of default, helping lenders prioritize monitoring and adjust terms for those under financial strain.

In [97]:
```
query = """ WITH seg AS (
              SELECT
                CASE
                  WHEN dti < 10 THEN '<10'
                  WHEN dti >=10 AND dti < 20 THEN '10-20'
                  WHEN dti >= 20 AND dti < 30 THEN '20-30'
                  WHEN dti >= 30 AND dti < 40 THEN '30-40'
                  ELSE '>=40'
                END AS dti_bucket,
                CASE
                  WHEN revol_util < 30 THEN '<30%'
                  WHEN revol_util >= 30 AND revol_util < 60 THEN '30-60%'
                  WHEN revol_util >= 60 AND revol_util < 90 THEN '60-90%'
                  ELSE '>=90%'
                END AS revol_util_bucket,
                loan_status,
                int_rate
              FROM loans
            )

            SELECT
              dti_bucket,
              revol_util_bucket,
              COUNT(*) AS loan_count,
              ROUND(AVG(CASE WHEN loan_status = 'charged off' THEN 1 ELSE 0 END)*100, 2) AS default_pct,
              ROUND(AVG(int_rate), 2) AS avg_int_rate
            FROM seg
            GROUP BY dti_bucket, revol_util_bucket
            ORDER BY dti_bucket , loan_count DESC;"""

df = run_query(conn,query)
df['default_pct'] = df['default_pct'].astype('float')
df['avg_int_rate'] = df['avg_int_rate'].astype('float')
df
```

Out[97]:

	dti_bucket	revol_util_bucket	loan_count	default_pct	avg_int_rate
0	<10	30-60%	7623	8.38	11.11
1	<10	<30%	6665	7.41	10.26
2	<10	60-90%	4504	8.10	12.03
3	<10	>=90%	904	9.51	12.80
4	>=40	60-90%	794	20.40	18.86
5	>=40	30-60%	587	15.84	17.55
6	>=40	>=90%	119	21.01	19.70
7	>=40	<30%	104	19.23	15.90
8	10-20	30-60%	20896	9.68	11.40
9	10-20	60-90%	15476	10.22	12.46
10	10-20	<30%	10231	9.15	10.49
11	10-20	>=90%	3011	11.29	13.26
12	20-30	30-60%	16075	12.78	12.42
13	20-30	60-90%	13755	12.82	13.59
14	20-30	<30%	6080	11.74	11.41
15	20-30	>=90%	2713	13.93	14.43
16	30-40	60-90%	6215	17.10	15.94
17	30-40	30-60%	6135	17.33	14.99
18	30-40	<30%	1820	16.98	13.89
19	30-40	>=90%	1168	17.21	16.43

In [98]:
```
default_pct_pivot = df.pivot(index = 'dti_bucket', columns = 'revol_util_bucket', values = 'default_pct')
avg_rate_pivot = df.pivot(index = 'dti_bucket', columns = 'revol_util_bucket', values = 'avg_int_rate')

fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize=(18,8))


sns.heatmap(default_pct_pivot, annot = True, fmt = "1.2f", cmap="inferno", ax=ax[0])
ax[0].set_title("Default % by DTI and Revolving Utilization", fontsize = 14, fontweight = 'bold')
ax[0].set_xlabel("Revolving Utilization Bucket")
ax[0].set_ylabel("DTI Bucket")

sns.heatmap(avg_rate_pivot, annot = True, fmt = "1.2f", cmap = "jet", ax = ax[1])
ax[1].set_title("Average Interest Rate by DTI and Revolving Utilization", fontsize = 14, fontweight = 'bold')
ax[1].set_xlabel("Revolving Utilization Bucket")
ax[1].set_ylabel("DTI Bucket")

plt.tight_layout()
plt.show()
```



35. Emp-Length Risk Ranking (by delinquency & default). Verifies if tenure stability should reduce delinquencies and defaults.

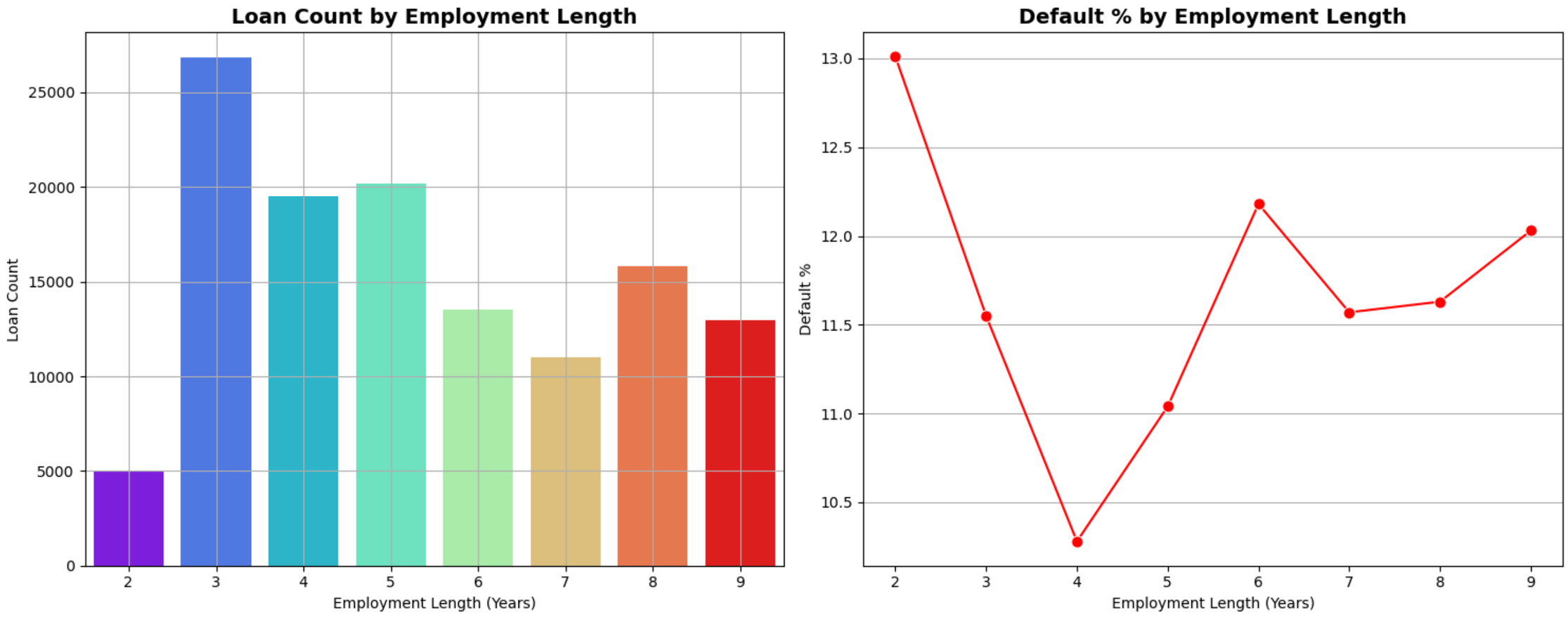
The analysis shows that employment length has a clear relationship with loan default risk. Borrowers with very short employment tenure (2–3 years) have the highest default percentages, while those with longer tenures (7–9 years) tend to default less, indicating that job stability reduces repayment risk. Loan counts are highest among mid-tenure borrowers (3–8 years), showing that most borrowers fall in this range, but defaults still vary based on tenure. The line plot highlights a gradual decrease in default rates as employment length increases, supporting the idea that stable employment provides a protective effect. Overall, lenders can use employment length as a simple yet effective signal to rank borrower risk and adjust lending strategies accordingly.

In [99]:
```
query = """ WITH m AS (
              SELECT
                b.emp_length AS 'emp_len',
                COUNT(*) AS 'loan_count',
                AVG(l.delinq_2yrs) AS 'avg_delinq_2yrs',
                AVG(CASE WHEN l.loan_status = 'charged off' THEN 1 ELSE 0 END) AS 'default_rate'
              FROM borrowers b
              JOIN loans l
              ON l.borrower_id = b.borrower_id
              GROUP BY b.emp_length
              HAVING COUNT(*) >= 100
            )
            SELECT
              emp_len,
              loan_count,
              ROUND(avg_delinq_2yrs, 2) AS 'avg_delinq_2yrs',
              ROUND(default_rate*100, 2) AS 'default_pct',
              RANK() OVER (ORDER BY default_rate DESC) AS risk_rank
            FROM m
            ORDER BY risk_rank;"""

df = run_query(conn,query)
df
```

Out[99]:

	emp_len	loan_count	avg_delinq_2yrs	default_pct	risk_rank
0	2	5033	0.33	13.01	1
1	6	13551	0.34	12.18	2
2	9	12947	0.34	12.03	3
3	8	15826	0.31	11.63	4
4	7	11011	0.31	11.57	5
5	3	26827	0.37	11.55	6
6	5	20169	0.37	11.04	7
7	4	19511	0.37	10.28	8

In [100…
```
fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize = (15,6))

sns.barplot(data = df,x ='emp_len',hue ='emp_len',y ='loan_count',palette = 'rainbow',ax = ax[0],legend = False)
ax[0].set_xlabel("Employment Length (Years)")
ax[0].set_ylabel("Loan Count")
ax[0].set_title("Loan Count by Employment Length", fontsize = 14, fontweight = 'bold')
ax[0].grid(True)

sns.lineplot(data = df,x = 'emp_len',y = 'default_pct',marker = 'o',color = 'red',markersize = 8,ax = ax[1])
ax[1].set_xlabel("Employment Length (Years)")
ax[1].set_ylabel("Default %")
ax[1].set_title("Default % by Employment Length", fontsize = 14, fontweight = 'bold')
ax[1].grid(True, axis='y')

plt.tight_layout()
plt.show()
```



36. Home-Ownership Bias After Controlling for Grade. Do certain home-ownership types get higher/lower rates vs their grade average?

The analysis shows subtle patterns in how home-ownership affects loan terms within the same grade. For example, in grade F, borrowers who own homes had slightly higher interest rates (+0.05% vs grade average) while renters were slightly below average (-0.02%), indicating lenders may see owners as able to handle higher risk. Mortgage holders often took larger loans, like in grade G where their average loan amount was $19,062, about $465 above the grade average, while renters had smaller loans ($18,485, $112 below average). Across most grades, the differences are modest, showing that grade remains the main driver, but home-ownership subtly adjusts both interest rate and loan size. Overall, lenders appear to make small, data-driven tweaks based on property status to balance risk while keeping terms consistent with the grade.

In [101…

```
query = """ WITH base AS (
              SELECT l.grade, b.home_ownership, l.int_rate, l.loan_amnt
              FROM loans l
              JOIN borrowers b
              ON b.borrower_id = l.borrower_id
            ),
            grade_avg AS (
              SELECT grade,
                     ROUND(AVG(int_rate),2)  AS 'grade_avg_rate',
                     ROUND(AVG(loan_amnt)) AS 'grade_avg_amt'
              FROM base
              GROUP BY grade
            )
            SELECT
              b.grade,
              b.home_ownership,
              COUNT(*) AS loan_count,g.grade_avg_rate, g.grade_avg_amt,
              ROUND(AVG(b.int_rate), 2) AS 'avg_int_rate',
              ROUND(AVG(b.int_rate) - g.grade_avg_rate, 2) AS 'rate_vs_grade',
              ROUND(AVG(b.loan_amnt)) AS 'avg_loan_amnt',
              ROUND(AVG(b.loan_amnt) - g.grade_avg_amt, 0) AS 'amt_vs_grade'
            FROM base b
            JOIN grade_avg g
            ON b.grade = g.grade
            GROUP BY b.grade, b.home_ownership, g.grade_avg_rate, g.grade_avg_amt
            HAVING loan_count >= 100
            ORDER BY b.grade, rate_vs_grade DESC;"""

df = run_query(conn,query)
df
```

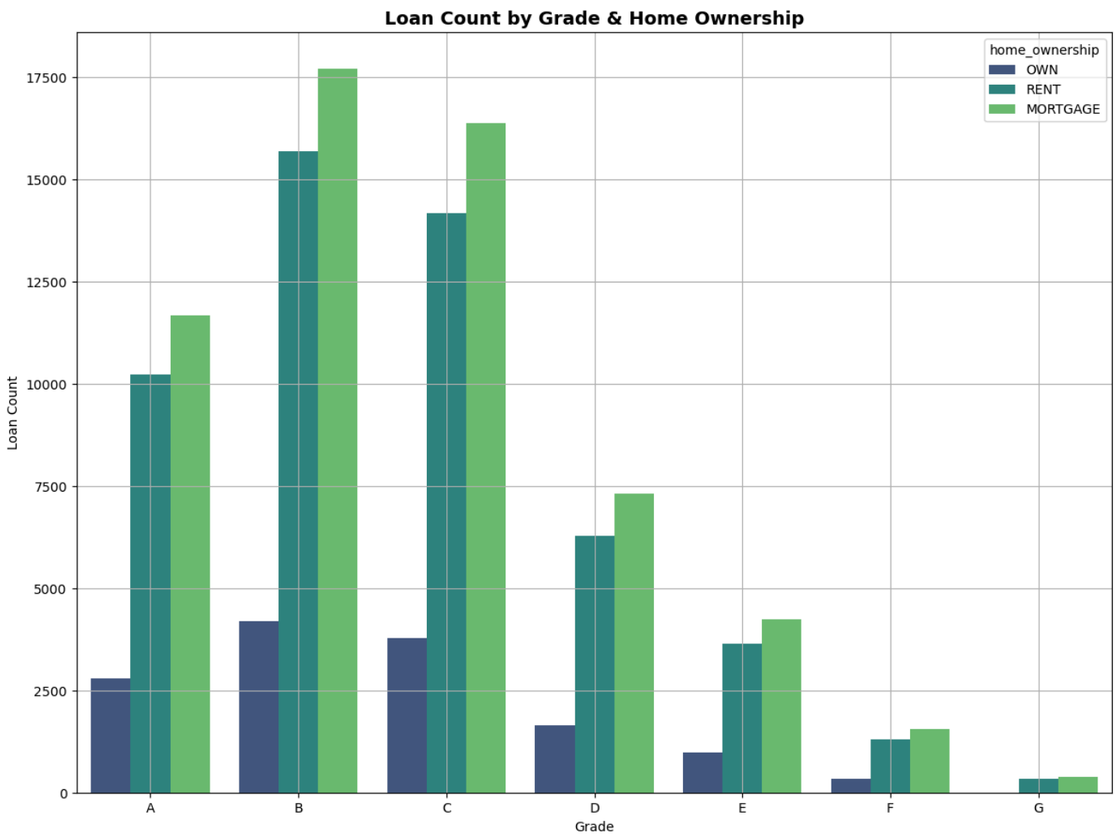
Out[101…

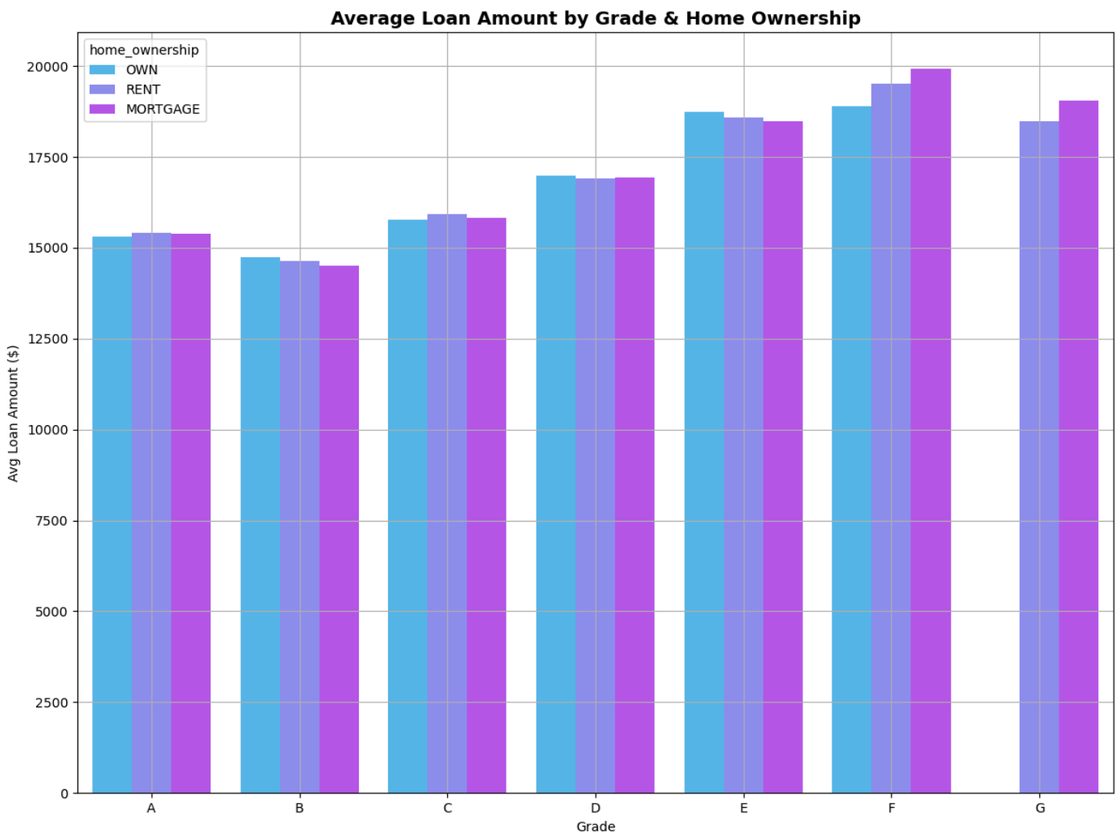
	grade	home_ownership	loan_count	grade_avg_rate	grade_avg_amt	avg_int_rate	rate_vs_grade	avg_loan_amnt	amt_vs_grade
0	A	OWN	2796	6.68	15398	6.69	0.01	15313	-85
1	A	RENT	10242	6.68	15398	6.69	0.01	15423	25
2	A	MORTGAGE	11687	6.68	15398	6.67	-0.01	15397	-1
3	B	RENT	15687	9.97	14586	9.98	0.01	14640	54
4	B	OWN	4194	9.97	14586	9.96	-0.01	14745	159
5	B	MORTGAGE	17717	9.97	14586	9.96	-0.01	14501	-85
6	C	MORTGAGE	16375	13.47	15863	13.47	0.00	15827	-36
7	C	RENT	14172	13.47	15863	13.46	-0.01	15930	67
8	C	OWN	3800	13.47	15863	13.45	-0.02	15771	-92
9	D	OWN	1665	17.60	16927	17.64	0.04	16989	62
10	D	MORTGAGE	7334	17.60	16927	17.60	0.00	16928	1
11	D	RENT	6280	17.60	16927	17.59	-0.01	16910	-17
12	E	OWN	989	20.74	18548	20.76	0.02	18736	188
13	E	MORTGAGE	4248	20.74	18548	20.75	0.01	18479	-69
14	E	RENT	3648	20.74	18548	20.72	-0.02	18577	29
15	F	OWN	358	24.29	19650	24.34	0.05	18899	-751
16	F	MORTGAGE	1559	24.29	19650	24.29	0.00	19928	278
17	F	RENT	1303	24.29	19650	24.27	-0.02	19524	-126
18	G	RENT	345	27.99	18597	28.04	0.05	18485	-112
19	G	MORTGAGE	391	27.99	18597	27.97	-0.02	19062	465

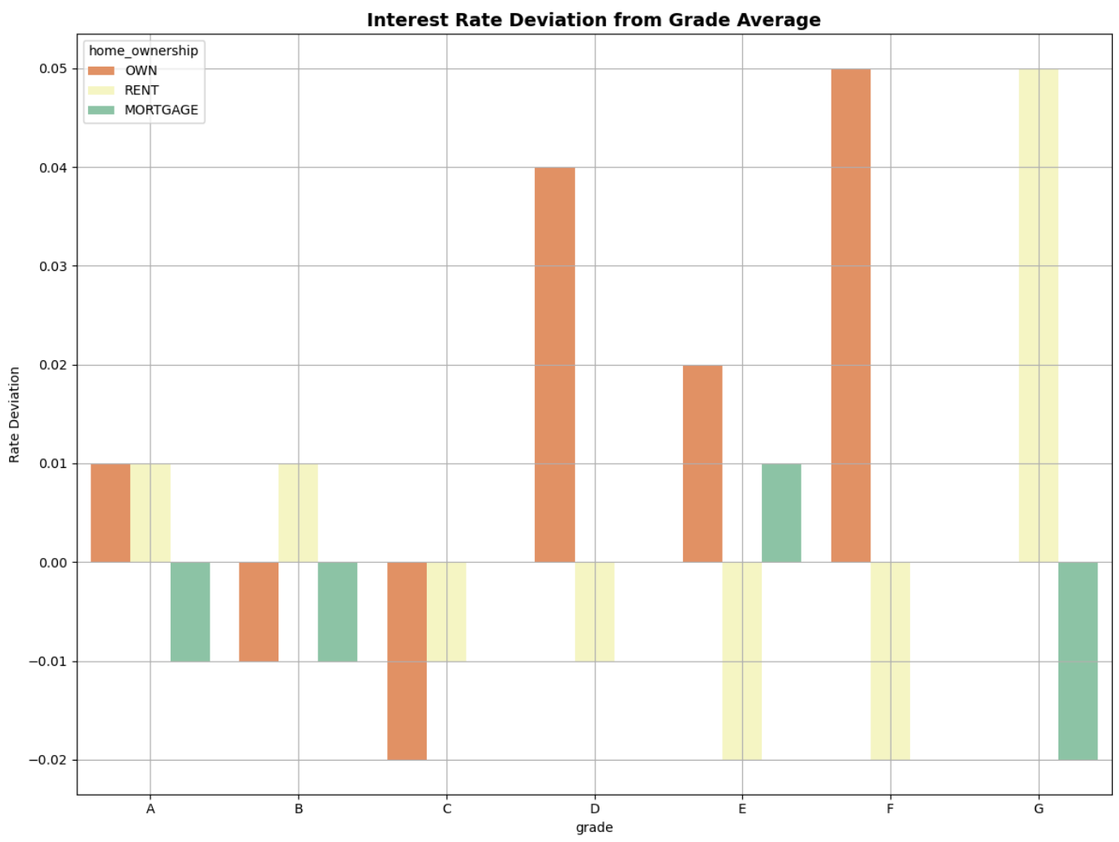
In [102…

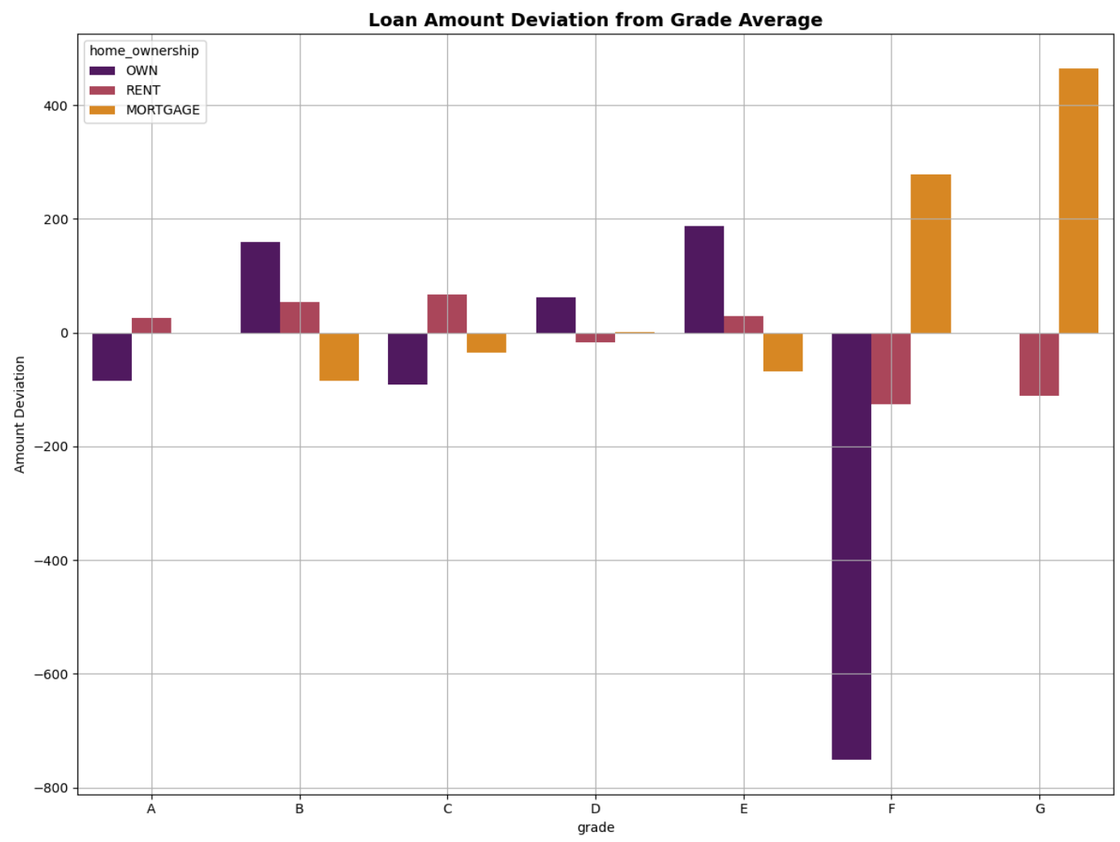
```
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(24,18))

sns.barplot(data = df,x = 'grade',y = 'loan_count',hue = 'home_ownership',palette = 'viridis',ax = ax[0,0])
ax[0,0].set_title('Loan Count by Grade & Home Ownership',fontsize = 14,fontweight = 'bold')
ax[0,0].set_ylabel('Loan Count')
ax[0,0].set_xlabel('Grade')
ax[0,0].grid(True)

sns.barplot(data = df,x = 'grade',y = 'avg_loan_amnt',hue = 'home_ownership',palette = 'cool',ax = ax[0,1])
ax[0,1].set_title('Average Loan Amount by Grade & Home Ownership',fontsize = 14,fontweight = 'bold')
ax[0,1].set_ylabel('Avg Loan Amount ($)')
```

```
ax[0,1].set_xlabel('Grade')
ax[0,1].grid(True)

sns.barplot(data = df,x = 'grade',y = 'rate_vs_grade',hue = 'home_ownership',palette = 'Spectral',ax = ax[1,0])
ax[1,0].set_title('Interest Rate Deviation from Grade Average',fontsize = 14,fontweight = 'bold')
ax[1,0].set_ylabel('Rate Deviation')
ax[1,0].grid(True)

sns.barplot(data = df,x = 'grade',y = 'amt_vs_grade',hue = 'home_ownership',palette = 'inferno',ax = ax[1,1])
ax[1,1].set_title('Loan Amount Deviation from Grade Average',fontsize = 14,fontweight = 'bold')
ax[1,1].set_ylabel('Amount Deviation')
ax[1,1].grid(True)

plt.tight_layout()
plt.show()
```









37. Concentration Risk: Top Grade-Purpose Mix by Share of Portfolio. Identify segments dominating exposure.

The concentration risk analysis shows that a few loan segments dominate the portfolio. Debt consolidation loans in grades C, B, and A alone make up over 43% of total exposure, with grade C debt consolidation topping the list at 17.4%. Credit card loans and lower-grade debt consolidation add another 15–20%, highlighting that a handful of grade–purpose combinations account for most of the portfolio. The cumulative share plot shows that the top 10 segments cover more than 82% of the portfolio, revealing a strong concentration in a few segments. This indicates potential vulnerability: if defaults spike in these top segments, portfolio health could be significantly affected. Lenders can use this insight to diversify exposure, adjust risk limits, or focus monitoring on high-share segments to reduce systemic risk.

In [110…

```
query = """WITH mix AS (
  SELECT grade, purpose, SUM(loan_amnt) AS amt
  FROM loans
  GROUP BY grade, purpose
),
tot AS (SELECT SUM(loan_amnt) AS total_amt FROM loans)
SELECT
  m.grade,
  m.purpose,
  m.amt,
  ROUND(m.amt / t.total_amt * 100, 2) AS portfolio_share_pct,
  RANK() OVER (ORDER BY m.amt DESC) AS size_rank
FROM mix m
CROSS JOIN tot t
ORDER BY size_rank LIMIT 15;"""

df = run_query(conn,query)
df
```

Out[110...

	grade	purpose	amt	portfolio_share_pct	size_rank
0	C	debt_consolidation	343177700	17.37	1
1	B	debt_consolidation	313765725	15.88	2
2	A	debt_consolidation	195335425	9.89	3
3	D	debt_consolidation	169281800	8.57	4
4	B	credit_card	161796075	8.19	5
5	A	credit_card	129509075	6.55	6
6	C	credit_card	121115850	6.13	7
7	E	debt_consolidation	113031300	5.72	8
8	F	debt_consolidation	45024625	2.28	9
9	D	credit_card	43373150	2.20	10
10	B	home_improvement	32014400	1.62	11
11	C	home_improvement	31447750	1.59	12
12	A	home_improvement	27670525	1.40	13
13	E	credit_card	22459400	1.14	14
14	C	other	20467650	1.04	15

In [111...

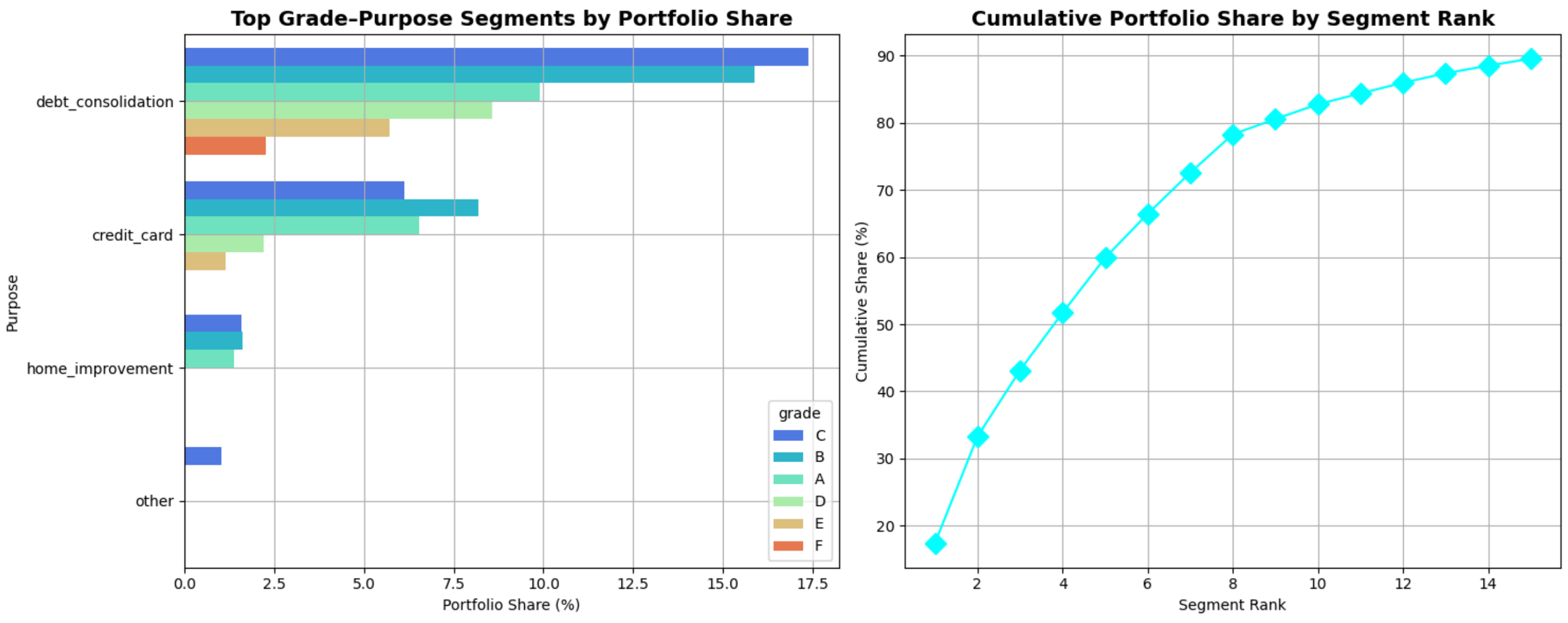
```
# Sort by portfolio share %
df_sorted = df.sort_values("portfolio_share_pct", ascending = False).reset_index(drop = True)
df_sorted["cum_share"] = df_sorted["portfolio_share_pct"].cumsum()
df_sorted
```

Out[111...

	grade	purpose	amt	portfolio_share_pct	size_rank	cum_share
0	C	debt_consolidation	343177700	17.37	1	17.37
1	B	debt_consolidation	313765725	15.88	2	33.25
2	A	debt_consolidation	195335425	9.89	3	43.14
3	D	debt_consolidation	169281800	8.57	4	51.71
4	B	credit_card	161796075	8.19	5	59.90
5	A	credit_card	129509075	6.55	6	66.45
6	C	credit_card	121115850	6.13	7	72.58
7	E	debt_consolidation	113031300	5.72	8	78.30
8	F	debt_consolidation	45024625	2.28	9	80.58
9	D	credit_card	43373150	2.20	10	82.78
10	B	home_improvement	32014400	1.62	11	84.40
11	C	home_improvement	31447750	1.59	12	85.99
12	A	home_improvement	27670525	1.40	13	87.39
13	E	credit_card	22459400	1.14	14	88.53
14	C	other	20467650	1.04	15	89.57

In [118...

```
fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize = (15,6))

# 1. Barplot: portfolio share % show top 15 for clarity
sns.barplot(data = df_sorted.head(15),x = "portfolio_share_pct",y = "purpose",hue = "grade",palette = "rainbow",ax = ax[0])
ax[0].set_title("Top Grade-Purpose Segments by Portfolio Share",fontsize = 14,fontweight = "bold")
ax[0].set_xlabel("Portfolio Share (%)")
ax[0].set_ylabel("Purpose")
ax[0].grid(True)

# 2. Line plot: cumulative portfolio share
ax[1].plot(df_sorted["size_rank"],df_sorted["cum_share"],marker = "D",color = "cyan",markersize = 10)
ax[1].set_title("Cumulative Portfolio Share by Segment Rank",fontsize = 14,fontweight = "bold")
ax[1].set_xlabel("Segment Rank")
ax[1].set_ylabel("Cumulative Share (%)")
ax[1].grid(True)

plt.tight_layout()
plt.show()
```



38. Cross-Grade Borrowers vs Single-Grade: Risk & Pricing. Multi-grade borrowers may indicate repeat borrowing; check risk & pricing.

The analysis shows that borrowers taking loans across multiple grades (multi-grade) are slightly riskier than single-grade borrowers, with a default rate of 11.6% versus 10.7%. Multi-grade borrowers also face higher average interest rates (12.6% compared to 11.9%), reflecting lenders' attempts to price in the additional risk. This suggests that repeat or varied borrowing behavior may indicate financial strain or over-leveraging. The bar plots clearly highlight that while single-grade borrowers are more stable,

multi-grade borrowers form a larger portion of total loans (102,547 vs 22,328), making them an important segment to monitor. Lenders can use this insight to adjust credit limits, interest rates, or implement stricter monitoring for multi-grade borrowers to mitigate potential portfolio risk.

In [119…

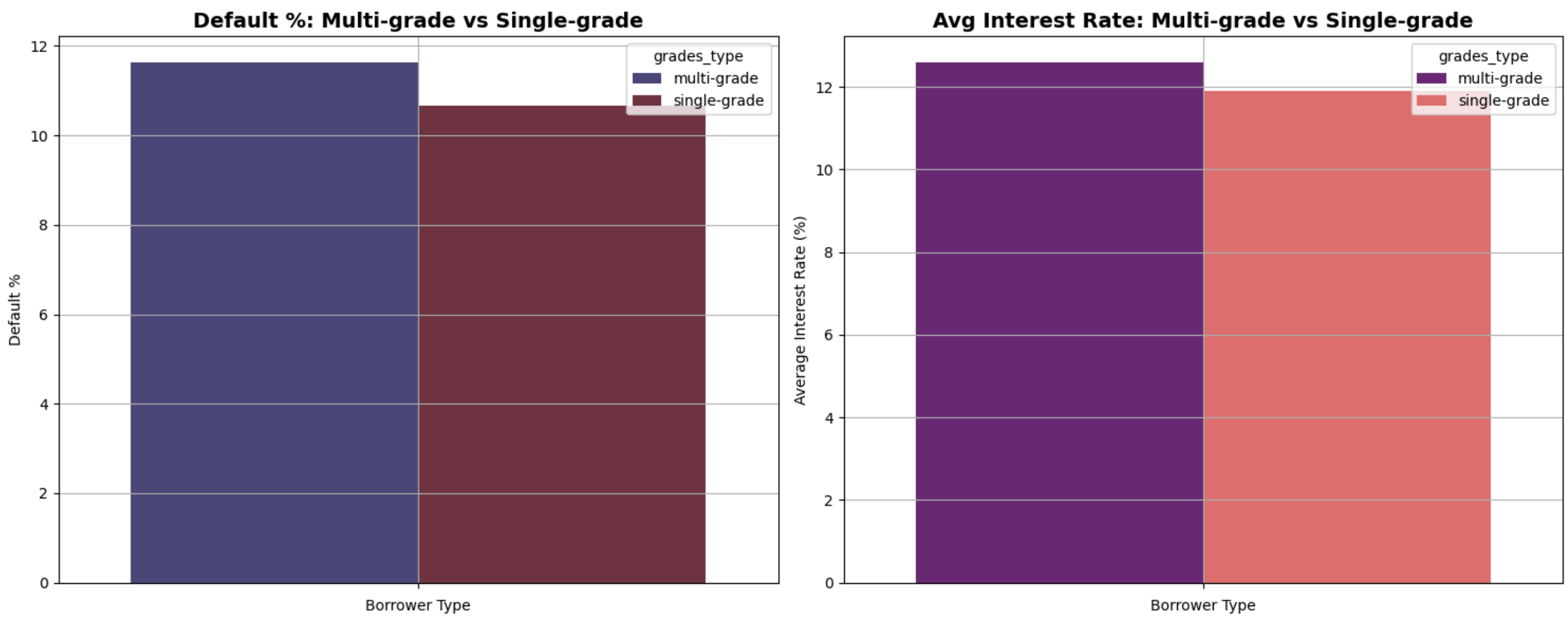
```sql
query = """ WITH grade_counts AS (
              SELECT borrower_id, COUNT(DISTINCT grade) AS distinct_grades
              FROM loans
              GROUP BY borrower_id
            ),
            labeled AS (
              SELECT
                CASE WHEN gc.distinct_grades >= 2 THEN 'multi-grade' ELSE 'single-grade' END AS grades_type,
                l.loan_status,
                l.int_rate
              FROM grade_counts gc
              JOIN loans l
              ON l.borrower_id = gc.borrower_id
            )

            SELECT
              grades_type,
              COUNT(*) AS loan_count,
              ROUND(AVG(CASE WHEN loan_status = 'charged off' THEN 1 ELSE 0 END)*100, 2) AS default_pct,
              ROUND(AVG(int_rate), 2) AS avg_int_rate
            FROM labeled
            GROUP BY grades_type;"""

df = run_query(conn,query)
df
```

Out[119…

	grades_type	loan_count	default_pct	avg_int_rate
0	multi-grade	102547	11.63	12.59
1	single-grade	22328	10.67	11.90

In [124…

```sql
fig, ax = plt.subplots(nrows = 1,ncols = 2, figsize = (15,6))

# 1. Default percentage
sns.barplot( data = df,hue = "grades_type",y = "default_pct",palette = "icefire",ax = ax[0])
ax[0].set_title("Default %: Multi-grade vs Single-grade",fontsize = 14,fontweight = "bold")
ax[0].set_xlabel("Borrower Type")
ax[0].set_ylabel("Default %")
ax[0].grid(True)

# 2. Average Interest Rate
sns.barplot(data = df,hue = "grades_type",y = "avg_int_rate",palette = "magma",ax = ax[1])
ax[1].set_title("Avg Interest Rate: Multi-grade vs Single-grade",fontsize = 14,fontweight = "bold")
ax[1].set_xlabel("Borrower Type")
ax[1].set_ylabel("Average Interest Rate (%)")
ax[1].grid(True)

plt.tight_layout()
plt.show()
```



39. Delinquency Signal by Grade: Impact Delta. Quantify how much a non-zero delinq_2yrs lifts default risk within each grade.

The analysis highlights how a recent delinquency (non-zero delinq_2yrs) impacts default risk across loan grades. Interestingly, borrowers with no prior delinquencies (clean) generally show much higher default rates than those with recorded delinquencies, with the difference growing from Grade A (delta = −1.95%) to Grade G (delta = −23.5%). This counterintuitive pattern suggests that in lower grades, prior delinquencies may already be accounted for in interest rates or lending terms, while clean borrowers could still be exposed to higher absolute default risk. The bar plots show that the impact of delinquency on default is largest in high-risk grades (E–G), emphasizing that even “clean” borrowers in these grades can be risky. Lenders can use this insight to refine early warning models and adjust monitoring strategies, focusing not only on prior delinquencies but also on grade-specific risk profiles.

In [125…

```sql
query = """WITH stats AS (
              SELECT
                grade,
                AVG(CASE WHEN delinq_2yrs > 0 AND loan_status = 'charged off' THEN 1 ELSE 0 END) AS def_with_delinq,
                AVG(CASE WHEN delinq_2yrs = 0 AND loan_status = 'charged off' THEN 1 ELSE 0 END) AS def_clean
              FROM loans
              GROUP BY grade
            )

            SELECT
              grade,
              ROUND(def_with_delinq * 100, 2) AS default_pct_with_delinq,
              ROUND(def_clean * 100, 2) AS default_pct_clean,
              ROUND((def_with_delinq - def_clean) * 100, 2) AS delta_default_pct,
              RANK() OVER (ORDER BY (def_with_delinq - def_clean) DESC) AS impact_rank
            FROM stats
            ORDER BY impact_rank;"""

df = run_query(conn,query)
df
```

Out[125...

	grade	default_pct_with_delinq	default_pct_clean	delta_default_pct	impact_rank
0	A	0.46	2.41	-1.95	1
1	B	1.78	5.32	-3.54	2
2	C	2.88	9.91	-7.03	3
3	D	4.61	14.82	-10.21	4
4	E	4.95	19.91	-14.96	5
5	F	6.30	25.84	-19.54	6
6	G	7.80	31.30	-23.50	7

In [128...

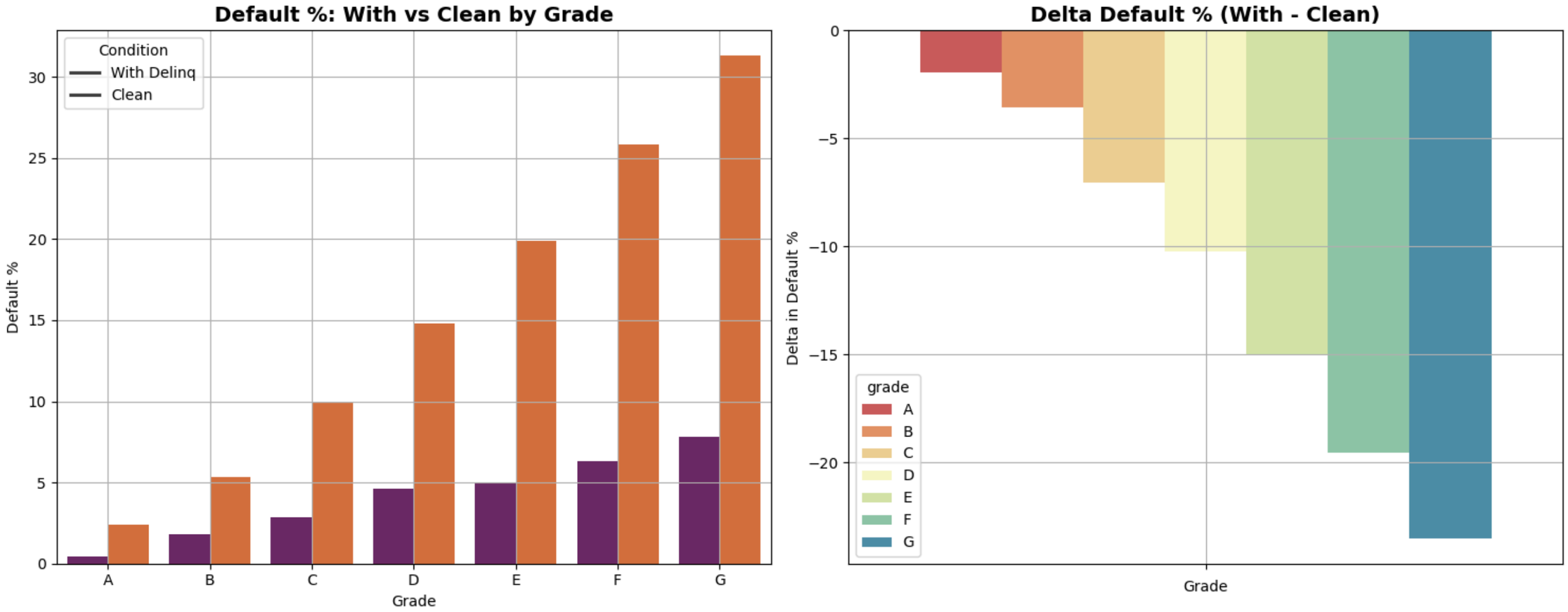
```
df_melt = df.melt(id_vars = "grade",value_vars = ["default_pct_with_delinq", "default_pct_clean"],
                  var_name = "condition",value_name = "default_pct")
df_melt
```

Out[128...

	grade	condition	default_pct
0	A	default_pct_with_delinq	0.46
1	B	default_pct_with_delinq	1.78
2	C	default_pct_with_delinq	2.88
3	D	default_pct_with_delinq	4.61
4	E	default_pct_with_delinq	4.95
5	F	default_pct_with_delinq	6.30
6	G	default_pct_with_delinq	7.80
7	A	default_pct_clean	2.41
8	B	default_pct_clean	5.32
9	C	default_pct_clean	9.91
10	D	default_pct_clean	14.82
11	E	default_pct_clean	19.91
12	F	default_pct_clean	25.84
13	G	default_pct_clean	31.30

In [133...

```
fig, ax = plt.subplots(1, 2, figsize = (15,6))

# With vs Clean default %
sns.barplot(data = df_melt,x = "grade",y = "default_pct",hue = "condition",palette = "inferno",ax = ax[0])
ax[0].set_title("Default %: With vs Clean by Grade", fontsize = 14, fontweight = "bold")
ax[0].set_xlabel("Grade")
ax[0].set_ylabel("Default %")
ax[0].legend(title = "Condition", labels = ["With Delinq", "Clean"])
ax[0].grid(True)

# Delta impact (difference)
sns.barplot(data = df,hue = "grade",y = "delta_default_pct",palette = "Spectral",ax = ax[1])
ax[1].set_title("Delta Default % (With - Clean)", fontsize = 14, fontweight = "bold")
ax[1].set_xlabel("Grade")
ax[1].set_ylabel("Delta in Default %")
ax[1].grid(True)

plt.tight_layout()
plt.show()
```



40. Safe Lending Zones (Best Performing Segments). Finds grade + purpose combinations with the highest repayment % . Identifies borrower groups where lending is most secure.

The analysis identifies the safest segments to lend to by combining loan grade and purpose. Grade A borrowers, especially for purposes like moving, house, and debt_consolidation, consistently show the highest repayment rates, with fully paid percentages reaching over 50% in some cases (e.g., moving loans at 50.6%). Grade B and C borrowers performing home improvement or debt consolidation loans also maintain decent repayment, typically around 33–37%. The heatmap clearly shows that higher-grade loans cluster in the top repayment zones, while lower grades (E–G) often have scattered and lower repayment percentages.Looking at loan volume versus repayment, larger segments like Grade A debt consolidation (12,170 loans) and Grade B debt consolidation (20,684 loans) dominate the portfolio but still maintain solid repayment, indicating both safety and portfolio scale. The boxplots and scatterplots reveal that high repayment percentages are concentrated in top grades, suggesting that lenders can focus on these safe lending zones to minimize risk while maintaining lending scale. Overall, targeting high-grade borrowers and proven purposes ensures portfolio stability and reduces default exposure.

In [136...

```
query = """WITH seg AS (
            SELECT
                grade,
                purpose,
                CASE
                    WHEN loan_status = 'fully paid' THEN 1
                    ELSE 0
                END AS fully_paid_flag,
                loan_id
```

```
            FROM loans
        )

        SELECT
            grade,
            purpose,
            COUNT(loan_id) AS total_loans,
            ROUND(AVG(fully_paid_flag) * 100, 2) AS fully_paid_pct
        FROM seg
        GROUP BY grade, purpose
        ORDER BY grade,fully_paid_pct DESC;"""

df = run_query(conn,query)
df['fully_paid_pct'] = df['fully_paid_pct'].astype('float')
df
```

Out[136...

	grade	purpose	total_loans	fully_paid_pct
0	A	moving	79	50.63
1	A	house	49	44.90
2	A	car	333	39.64
3	A	vacation	91	38.46
4	A	other	936	38.03
...	...	...	...	...
79	G	house	24	20.83
80	G	small_business	29	20.69
81	G	medical	7	14.29
82	G	credit_card	63	12.70
83	G	vacation	4	0.00

84 rows × 4 columns

In [137...

```
pivot_df1 = df.pivot_table(index = "grade",columns = "purpose",values = "total_loans")
pivot_df1
```

Out[137...

purpose grade	car	credit_card	debt_consolidation	home_improvement	house	major_purchase	medical	moving	other	renewable_energy	small_business	vacation
A	333.0	8167.0	12170.0	1896.0	49.0	704.0	181.0	79.0	936.0	6.0	113.0	91.0
B	379.0	10638.0	20684.0	2308.0	88.0	756.0	394.0	164.0	1720.0	14.0	263.0	190.0
C	298.0	7166.0	20716.0	2049.0	133.0	704.0	421.0	234.0	1981.0	22.0	378.0	245.0
D	142.0	2452.0	9540.0	962.0	104.0	310.0	200.0	131.0	1049.0	18.0	263.0	108.0
E	58.0	1173.0	5899.0	501.0	81.0	180.0	90.0	71.0	579.0	16.0	196.0	41.0
F	8.0	301.0	2227.0	197.0	44.0	63.0	23.0	30.0	211.0	5.0	102.0	9.0
G	7.0	63.0	528.0	48.0	24.0	16.0	7.0	11.0	83.0	1.0	29.0	4.0

In [138...

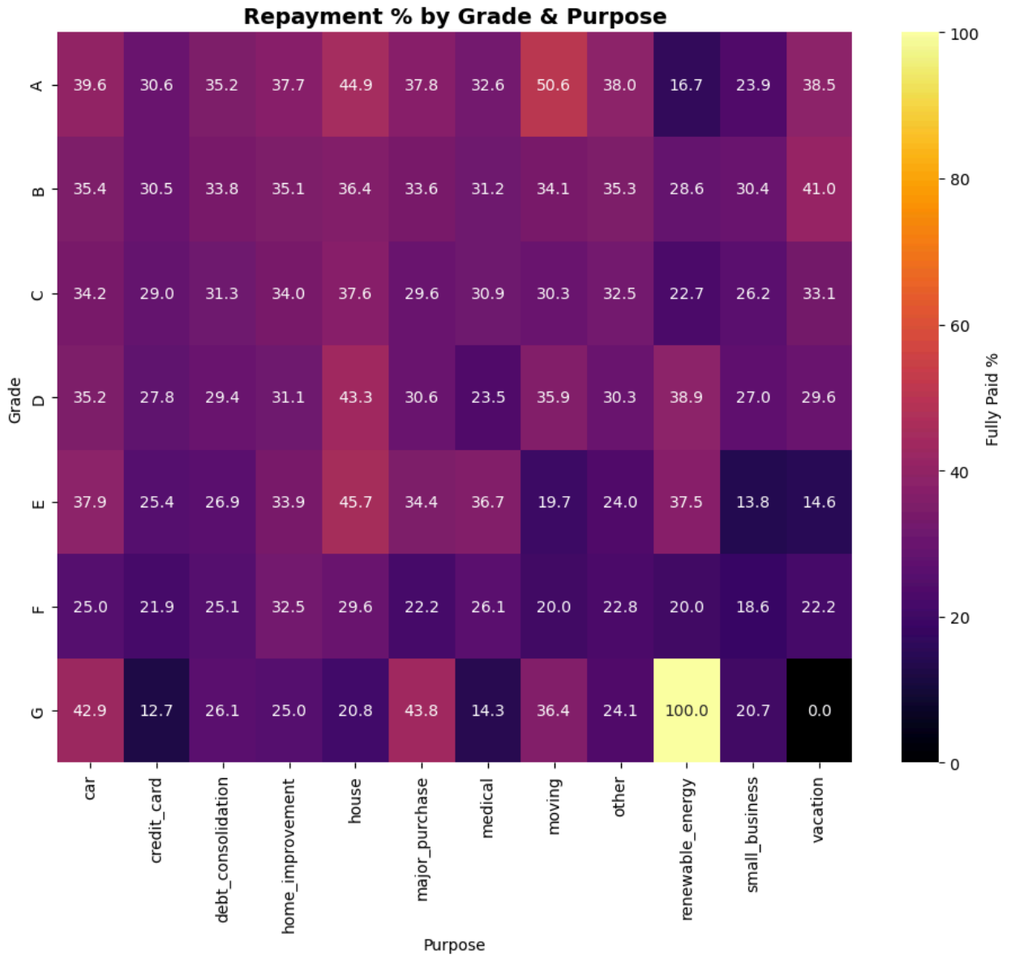
```
pivot_df2 = df.pivot_table(index = "grade",columns = "purpose",values = "fully_paid_pct")
pivot_df2
```

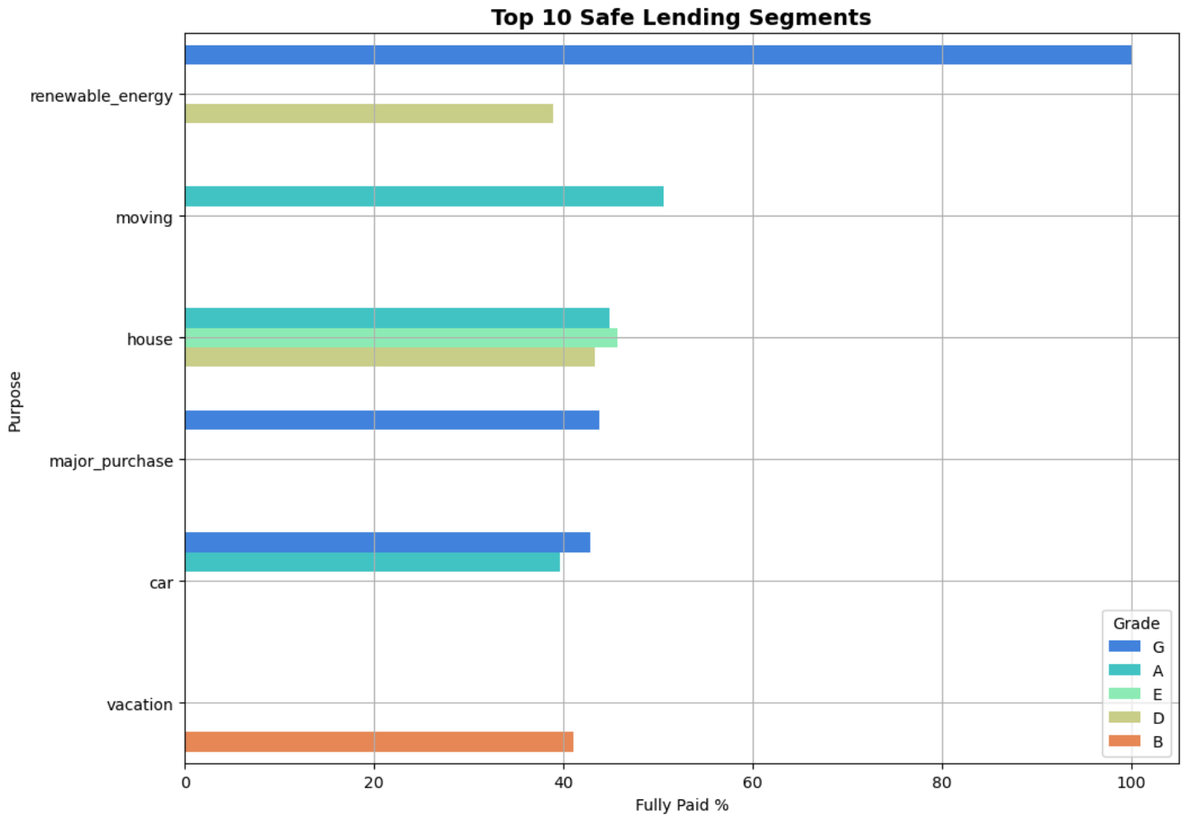
Out[138...

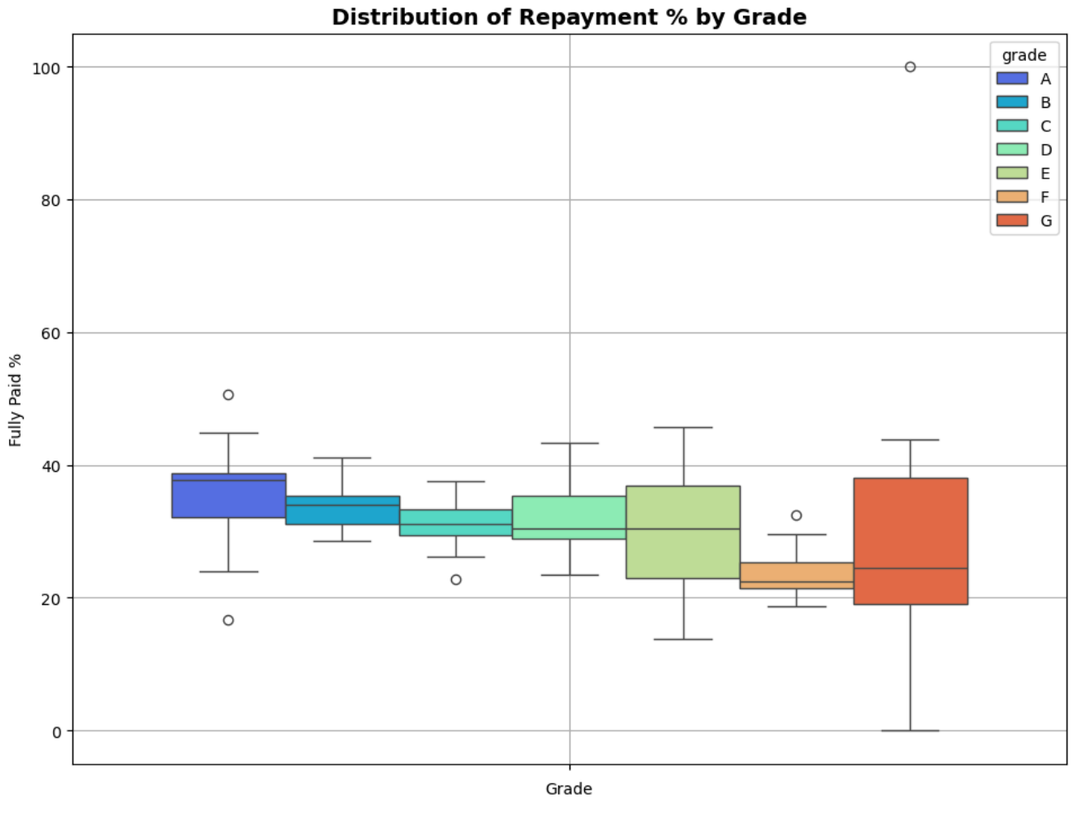
purpose grade	car	credit_card	debt_consolidation	home_improvement	house	major_purchase	medical	moving	other	renewable_energy	small_business	vacation
A	39.64	30.55	35.19	37.71	44.90	37.78	32.60	50.63	38.03	16.67	23.89	38.46
B	35.36	30.46	33.77	35.10	36.36	33.60	31.22	34.15	35.29	28.57	30.42	41.05
C	34.23	29.01	31.33	34.02	37.59	29.55	30.88	30.34	32.46	22.73	26.19	33.06
D	35.21	27.81	29.36	31.08	43.27	30.65	23.50	35.88	30.31	38.89	27.00	29.63
E	37.93	25.40	26.94	33.93	45.68	34.44	36.67	19.72	24.01	37.50	13.78	14.63
F	25.00	21.93	25.10	32.49	29.55	22.22	26.09	20.00	22.75	20.00	18.63	22.22
G	42.86	12.70	26.14	25.00	20.83	43.75	14.29	36.36	24.10	100.00	20.69	0.00

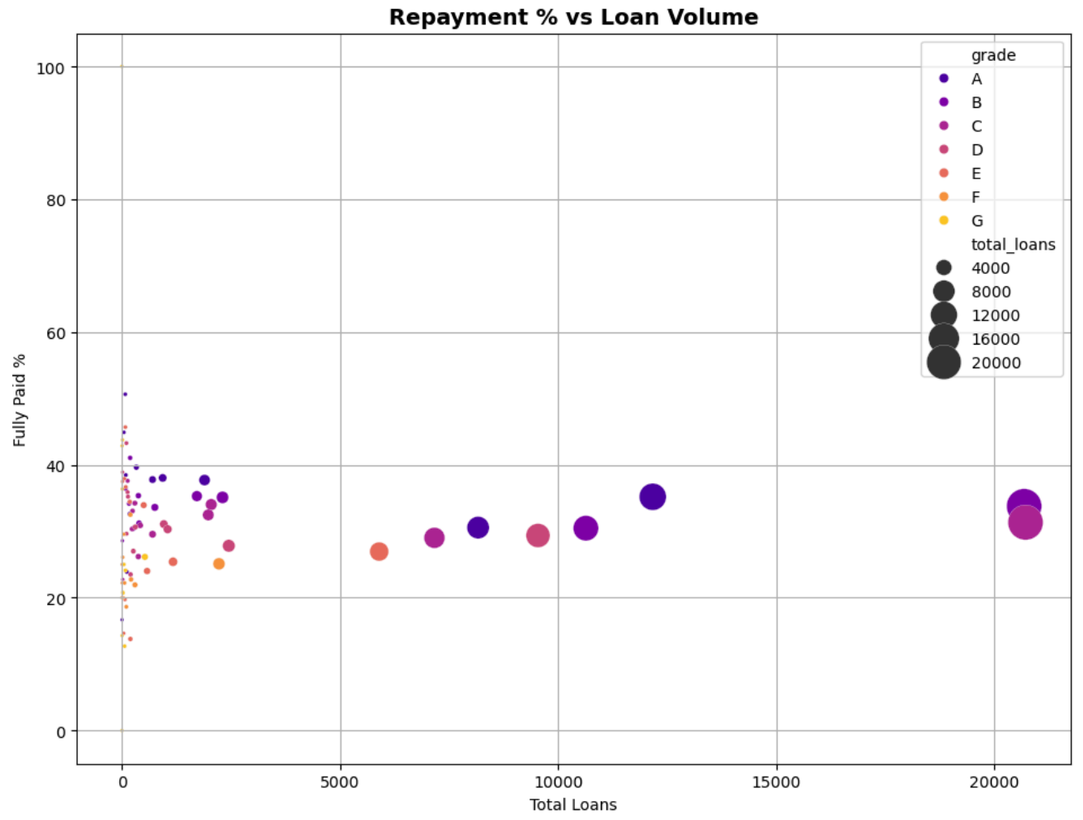
In [152...

```
fig, ax = plt.subplots(2, 2, figsize = (20,16))

# 1. Heatmap of repayment %
sns.heatmap(pivot_df2,annot = True, fmt = ".1f", cmap = "inferno",cbar_kws = {'label': 'Fully Paid %'},ax = ax[0,0])
ax[0,0].set_title("Repayment % by Grade & Purpose",fontsize = 14,fontweight = "bold")
ax[0,0].set_xlabel("Purpose")
ax[0,0].set_ylabel("Grade")

# 2. Top 10 segments barplot
top10 = df.sort_values("fully_paid_pct", ascending = False).head(10)

sns.barplot(data = top10,x = "fully_paid_pct",y = "purpose",hue = "grade",palette = "rainbow",ax = ax[0,1])
ax[0,1].set_title("Top 10 Safe Lending Segments", fontsize = 14, fontweight = "bold")
ax[0,1].set_xlabel("Fully Paid %")
ax[0,1].set_ylabel("Purpose")
ax[0,1].grid(True)
ax[0,1].legend(title = "Grade")

# 3. Boxplot repayment % by grade
sns.boxplot(data = df,hue = "grade",y="fully_paid_pct",palette = "rainbow",ax = ax[1,0])
ax[1,0].set_title("Distribution of Repayment % by Grade",fontsize = 14,fontweight = "bold")
ax[1,0].set_xlabel("Grade")
ax[1,0].set_ylabel("Fully Paid %")
ax[1,0].grid(True)

# 4. Scatterplot: total loans vs repayment %
sns.scatterplot(data = df,x = "total_loans",y = "fully_paid_pct",hue = "grade",palette = 'plasma',
                size = "total_loans",sizes = (5,500),ax=ax[1,1])
ax[1,1].set_title("Repayment % vs Loan Volume",fontsize = 14,fontweight = "bold")
ax[1,1].set_xlabel("Total Loans")
ax[1,1].set_ylabel("Fully Paid %")
ax[1,1].grid(True)

plt.tight_layout()
plt.show()
```









In [ ]:

In [ ]:

PROJECT SUMMARY

This project analyzed a large lending dataset combining borrower profiles, loan characteristics, and repayment behavior to uncover patterns, assess risk, and recommend actionable strategies for lenders. The goal was to understand who the bank lends to, how loans perform, which factors drive defaults, and how to optimize portfolio health.

1. Borrower Profile Analysis

- Borrowers are primarily middle-income individuals earning between $50k–$100k, with most holding 3–9 years of employment. This suggests a relatively stable mid-career population.
- Home ownership is concentrated in mortgages (~50%) and rentals (~41%), while verified borrowers make up about two-thirds of the population. Verification improves lending confidence but is not a perfect predictor of repayment.
- Borrowers often have multiple loans and high total credit accounts, signaling potential over-leverage risk.
- Key insight: Employment stability, home ownership, and verification combined with credit grade are stronger predictors of repayment than income alone.

2. Loan Portfolio Overview

- Loans mostly range from $1,000 to $40,000, with 36–60 month terms and average interest rates of 12.5%, reflecting risk-based pricing.
- Debt consolidation and credit card refinancing dominate, both in number and total amount, while niche purposes (home improvement, small business, renewable energy) contribute minimally.
- Grades B and C dominate the portfolio, while higher-risk grades (F and G) are smaller but carry higher interest rates.
- Key insight: Portfolio is moderately diversified but concentrated in a few segments. Longer-term and higher-purpose loans tend to carry higher risk and rates, requiring careful monitoring.

3. Loan Performance & Risk Analysis

- Majority of loans are current or fully paid, but risk clusters in mid-grade, short-term loans for high-risk purposes.
- Borrowers with past delinquencies, high DTI, or high revolving utilization are far more likely to default.
- Grade migration shows high-grade borrowers often improve or maintain status, while mid- and low-grade borrowers frequently deteriorate, signaling the need for ongoing credit monitoring.
- Interest rate analysis confirms lenders price risk: higher rates correlate with higher default, particularly for weaker borrowers.

4. Borrower Behavior & Repayment Patterns

- Stable employment and home ownership correlate with lower default rates. Renters and short-tenure employees are riskier.
- Repeat borrowing and multi-loan behavior indicate financial stress, increasing default likelihood.
- Borrower past behavior is predictive: first loan charged off → higher probability of future default; first loan fully paid → good repayment habit likely to continue.
- Key insight: Combining borrower traits, financial habits, and loan history is critical to predict defaults effectively.

5. Advanced Portfolio Insights & Strategic Recommendations

- High DTI and revolving utilization significantly increase default risk; employment length slightly influences risk; home ownership influences rates modestly.
- Debt consolidation in grades B–C dominates exposure, requiring active monitoring.
- Multi-grade borrowers exhibit slightly higher defaults and interest rates, signaling repeat borrowing risk.

- Delinquency history is the strongest early warning signal across all grades.
- Safe lending zones: high-grade borrowers in common purposes (debt consolidation, moving, home improvement) offer high repayment percentages with scale.

In [ ]:

STRATEGIC RECOMMENDATIONS FOR LENDERS

> *Focus on high-grade, stable borrowers for low-risk lending.*

> *Monitor mid- and low-grade borrowers intensively using multi-factor risk assessment.*

> *Track past delinquencies and repeat borrowers to flag early risks.*

> *Diversify portfolio beyond debt consolidation and credit cards to reduce concentration risk.*

> *Use adaptive interest rates and credit limits based on borrower behavior and repayment patterns.*

> *Implement early interventions: debt counseling, repayment incentives, or restructuring for high-risk borrowers.*

> *Build integrated scoring models combining grade, DTI, delinquencies, employment, and utilization to improve predictive accuracy.*

In [ ]:

OVERALL BUSSINESS INSIGHTS

> *Lending to high-grade, stable borrowers ensures portfolio stability.*

> *Mid-grade borrowers offer growth opportunities but require careful monitoring and dynamic risk-based strategies.*

> *Repeat borrowers, multi-grade exposure, high DTI, and prior delinquencies are key risk drivers.*

> *Behavior-driven adaptive strategies outperform static lending rules: early monitoring, risk-based pricing, and portfolio diversification significantly reduce defaults and improve returns.*

> *Practical takeaway for real-world lending: Lend to the stable, watch the risky, intervene early, and diversify smartly.*

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [109…
```
pip install mysql-connector-python sqlalchemy ipython-sql pandas matplotlib seaborn
```

```
Requirement already satisfied: mysql-connector-python in c:\users\divis\anaconda3\lib\site-packages (9.4.0)
Requirement already satisfied: sqlalchemy in c:\users\divis\anaconda3\lib\site-packages (2.0.39)
Requirement already satisfied: ipython-sql in c:\users\divis\anaconda3\lib\site-packages (0.5.0)
Requirement already satisfied: pandas in c:\users\divis\anaconda3\lib\site-packages (2.2.3)
Requirement already satisfied: matplotlib in c:\users\divis\anaconda3\lib\site-packages (3.10.0)
Requirement already satisfied: seaborn in c:\users\divis\anaconda3\lib\site-packages (0.13.2)
Requirement already satisfied: greenlet!=0.4.17 in c:\users\divis\anaconda3\lib\site-packages (from sqlalchemy) (3.1.1)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\divis\anaconda3\lib\site-packages (from sqlalchemy) (4.12.2)
Requirement already satisfied: prettytable in c:\users\divis\anaconda3\lib\site-packages (from ipython-sql) (3.16.0)
Requirement already satisfied: ipython in c:\users\divis\anaconda3\lib\site-packages (from ipython-sql) (8.30.0)
Requirement already satisfied: sqlparse in c:\users\divis\anaconda3\lib\site-packages (from ipython-sql) (0.5.3)
Requirement already satisfied: six in c:\users\divis\anaconda3\lib\site-packages (from ipython-sql) (1.17.0)
Requirement already satisfied: ipython-genutils in c:\users\divis\anaconda3\lib\site-packages (from ipython-sql) (0.2.0)
Requirement already satisfied: numpy>=1.26.0 in c:\users\divis\anaconda3\lib\site-packages (from pandas) (2.1.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\divis\anaconda3\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\divis\anaconda3\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\divis\anaconda3\lib\site-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\divis\anaconda3\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: decorator in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (5.1.1)
Requirement already satisfied: jedi>=0.16 in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (0.19.2)
Requirement already satisfied: matplotlib-inline in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (0.1.6)
Requirement already satisfied: prompt-toolkit<3.1.0,>=3.0.41 in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (3.0.43)
Requirement already satisfied: pygments>=2.4.0 in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (2.19.1)
Requirement already satisfied: stack-data in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (0.2.0)
Requirement already satisfied: traitlets>=5.13.0 in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (5.14.3)
Requirement already satisfied: colorama in c:\users\divis\anaconda3\lib\site-packages (from ipython->ipython-sql) (0.4.6)
Requirement already satisfied: wcwidth in c:\users\divis\anaconda3\lib\site-packages (from prompt-toolkit<3.1.0,>=3.0.41->ipython->ipython-sql) (0.2.5)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in c:\users\divis\anaconda3\lib\site-packages (from jedi>=0.16->ipython->ipython-sql) (0.8.4)
Requirement already satisfied: executing in c:\users\divis\anaconda3\lib\site-packages (from stack-data->ipython->ipython-sql) (0.8.3)
Requirement already satisfied: asttokens in c:\users\divis\anaconda3\lib\site-packages (from stack-data->ipython->ipython-sql) (3.0.0)
Requirement already satisfied: pure-eval in c:\users\divis\anaconda3\lib\site-packages (from stack-data->ipython->ipython-sql) (0.2.2)
Note: you may need to restart the kernel to use updated packages.
```

In [ ]: