BANK LOAN ANALYSIS USING MySQL QUERIES

Introduction on MySQL Analysis:

This analysis is based on a dataset containing detailed information on bank loan applications, disbursements, and repayments. The dataset was imported into MySQL from a .csv file and consists of thousands of records. Using MySQL queries, the data was cleaned, structured, and analyzed to understand key performance metrics such as loan volume trends, repayment patterns, disbursed amounts, interest rates, and risk segmentation. The objective of this analysis is to extract meaningful insights from the raw data that can guide better decision-making.

Created a Database called bank_loan_dB and then imported .csv file under the bank_loan_dB:

```
CREATE DATABASE bank_loan_dB;
```

Modified the Data Type of respective columns:

```
USE bank_loan_dB;

ALTER TABLE financial_loan

ADD PRIMARY KEY(id),

MODIFY COLUMN address_state VARCHAR(50),

MODIFY COLUMN application_type VARCHAR(50),

MODIFY COLUMN emp_length VARCHAR(50),

MODIFY COLUMN emp_title VARCHAR(100),

MODIFY COLUMN grade VARCHAR(50),

MODIFY COLUMN home_ownership VARCHAR(50),

MODIFY COLUMN loan_status VARCHAR(50),

MODIFY COLUMN purpose VARCHAR(50),

MODIFY COLUMN sub_grade VARCHAR(50),

MODIFY COLUMN term VARCHAR(50),

MODIFY COLUMN verification_status VARCHAR(50),

MODIFY COLUMN annual_income float,

MODIFY COLUMN dti float,

MODIFY COLUMN installment float,

MODIFY COLUMN int_rate float;
```

Modified the Text to Date Data Type using str_to_date function:

```
ALTER TABLE financial_loan;

UPDATE financial_loan

SET issue_date = str_to_date(issue_date, '%d-%m-%Y');

SET last_payment_date = str_to_date(last_payment_date, '%Y-%m-%d');

SET next_payment_date = str_to_date(next_payment_date, '%d-%m-%Y');
```

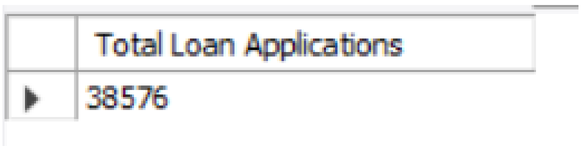
Column *last_payment_date* is in the format of %Y-%m-%d, it shows ERROR so created a new column and pasted all the values from the original column to the new column:

```
ALTER TABLE financial_loan

ADD COLUMN new_last_credit_pull_date date;

UPDATE financial_loan

SET new_last_credit_pull_date = str_to_date(last_credit_pull_date, '%d-%m-%Y');


ALTER TABLE financial_loan

MODIFY COLUMN issue_date date,

MODIFY COLUMN last_payment_date date,

MODIFY COLUMN next_payment_date date,

DROP COLUMN last_credit_pull_date,

CHANGE new_last_credit_pull_date last_credit_pull_date DATE;
```

WRITING QUERIES TO GET THE OVERVIEW OF THE BANK LOAN PERFORMANCE

Retrieved the total number of loan applications submitted:

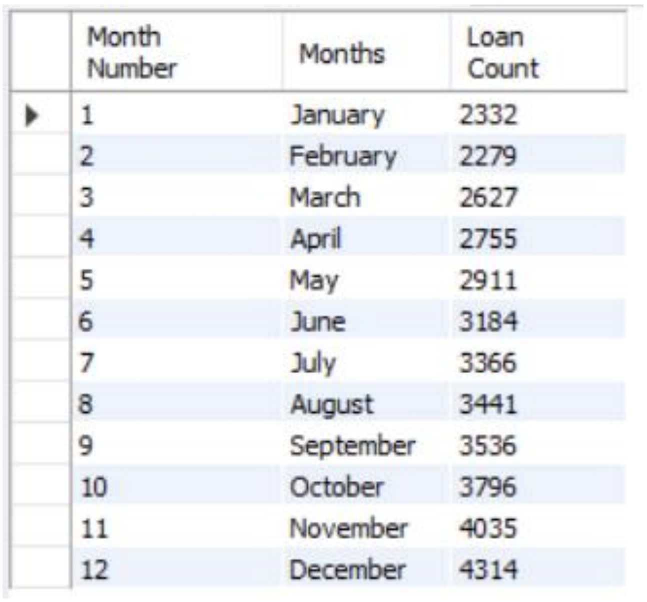
```
USE bank_loan_dB;

SELECT COUNT(id) as "Total Loan Applications" FROM financial_loan
```



Total Loan Applications
38576

Extracted the no. of loan applications submitted each month, ordered chronologically by month:

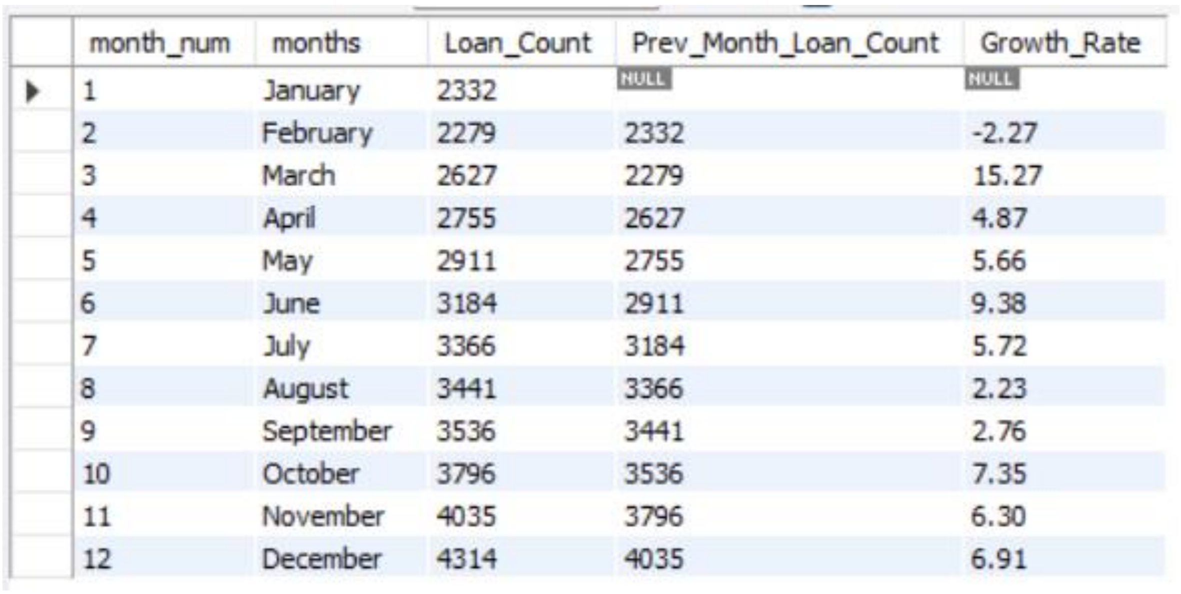
```
SELECT

        MONTH(issue_date) as "Month Number",

        MONTHNAME(issue_date) as "Months",

        COUNT(id) as "Loan Count"

FROM financial_loan

GROUP BY month(issue_date), monthname(issue_date)

ORDER BY month(issue_date);
```



Month Number	Months	Loan Count
1	January	2332
2	February	2279
3	March	2627
4	April	2755
5	May	2911
6	June	3184
7	July	3366
8	August	3441
9	September	3536
10	October	3796
11	November	4035
12	December	4314

Calculated the Month-over-Month (MoM) Growth Rate in loan volume:

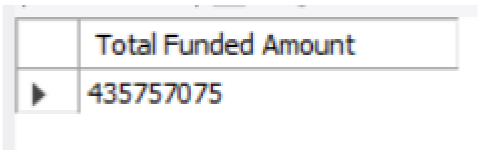
```
WITH MonthlyLoanData AS(
        SELECT
                MONTH(issue_date) as month_num,
                MONTHNAME(issue_date) as months,
                COUNT(id) as Loan_Count
        FROM financial_loan
        GROUP BY month(issue_date), monthname(issue_date)
        ORDER BY month(issue_date)
)
SELECT
        month_num,
        months,
        Loan_Count,
        LAG(Loan_Count) OVER (ORDER BY month_num) as Prev_Month_Loan_Count,
        ROUND(((Loan_Count - LAG(Loan_Count) OVER (ORDER BY month_num))/LAG(Loan_Count)
        OVER (ORDER BY month_num))*100,2) as Growth_Rate
FROM MonthlyLoanData
ORDER BY month_num;
```



month_num	months	Loan_Count	Prev_Month_Loan_Count	Growth_Rate
1	January	2332	NULL	NULL
2	February	2279	2332	-2.27
3	March	2627	2279	15.27
4	April	2755	2627	4.87
5	May	2911	2755	5.66
6	June	3184	2911	9.38
7	July	3366	3184	5.72
8	August	3441	3366	2.23
9	September	3536	3441	2.76
10	October	3796	3536	7.35
11	November	4035	3796	6.30
12	December	4314	4035	6.91

Computed the total disbursed amount for the year:

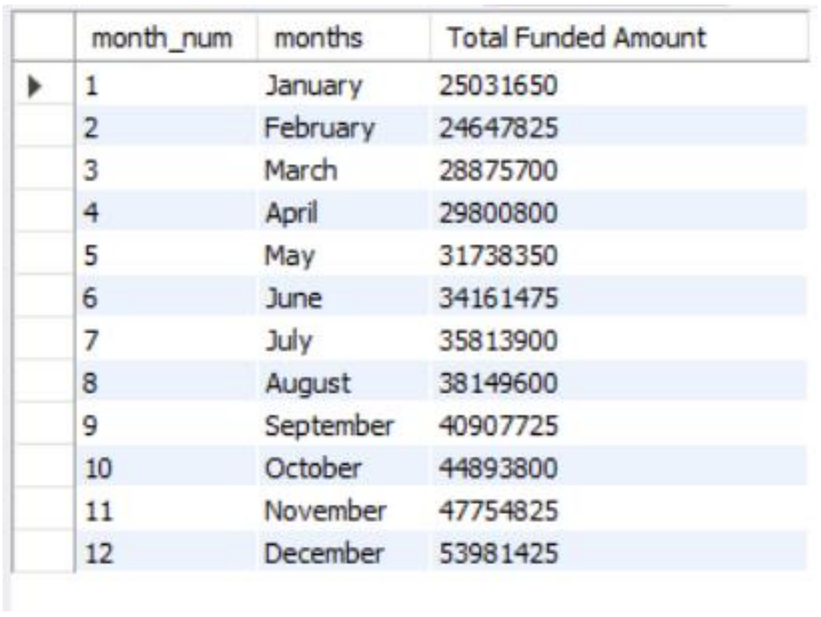
SELECT SUM(loan_amount) as "Total Funded Amount" FROM financial_loan;



Total Funded Amount
435757075

Displayed the disbursed amount on a monthly basis:

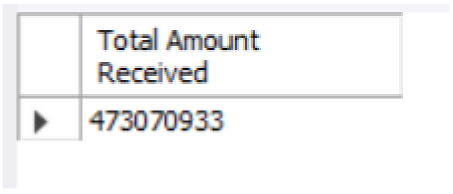
```
SELECT
        month(issue_date) as month_num,
        MONTHNAME(issue_date) as months,
        SUM(loan_amount) as "Total Funded Amount"
FROM financial_loan
GROUP BY month(issue_date), monthname(issue_date)
ORDER BY month(issue_date)
```



month_num	months	Total Funded Amount
1	January	25031650
2	February	24647825
3	March	28875700
4	April	29800800
5	May	31738350
6	June	34161475
7	July	35813900
8	August	38149600
9	September	40907725
10	October	44893800
11	November	47754825
12	December	53981425

Computed the total amount repaid by borrowers:

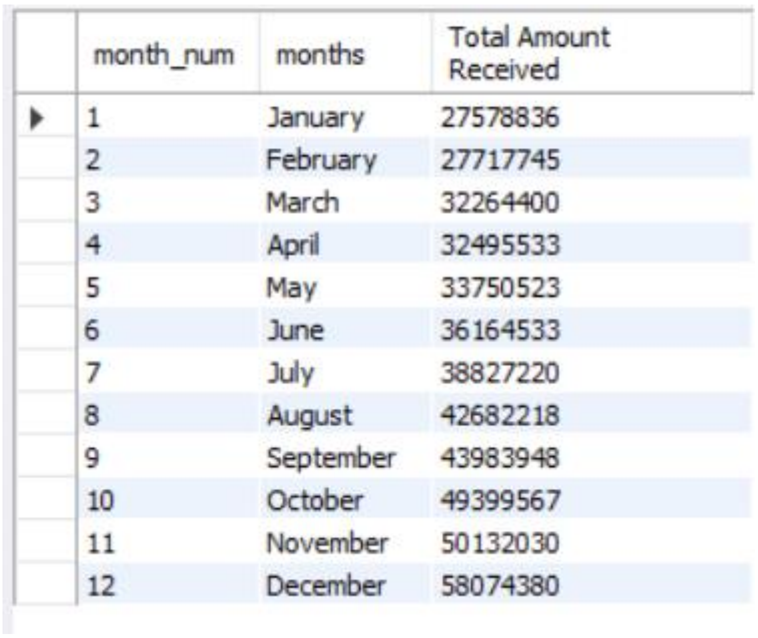
SELECT SUM(total_payment) as "Total Amount Received" FROM financial_loan



Total Amount Received
473070933

Computed the total amount received on a monthly basis:

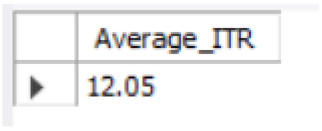
```
SELECT
        month(issue_date) as month_num,
        MONTHNAME(issue_date) as months,
        SUM(total_payment) as "Total Amount Received"
FROM financial_loan
GROUP BY month(issue_date), monthname(issue_date)
ORDER BY month(issue_date);
```



month_num	months	Total Amount Received
1	January	27578836
2	February	27717745
3	March	32264400
4	April	32495533
5	May	33750523
6	June	36164533
7	July	38827220
8	August	42682218
9	September	43983948
10	October	49399567
11	November	50132030
12	December	58074380

Calculate the average interest rate:

```
SELECT round(avg(int_rate)*100,2) as Average_ITR FROM financial_loan;
```

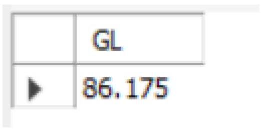


Average_ITR
12.05

Good Loans: Loans where borrowers repay regularly or have fully repaid; classified as *"Current"* or *"Fully Paid"* under loan status.

Analyzed the Percentage of Good Loans (GL):

```
SELECT
        ROUND((COUNT(CASE WHEN loan_status = "Fully Paid" OR loan_status = "Current" THEN id
        END))/COUNT(id)*100,3) as GL
FROM financial_loan;
```



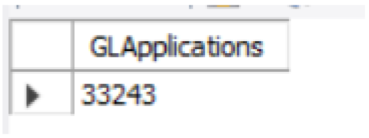
	GL
▶	86.175

Retrieved the Total Number of Good Loan Applications:

```
SELECT COUNT(CASE WHEN loan_status = "Fully Paid" OR loan_status = "Current" THEN id END) as
GL_Applications
FROM financial_loan;
```

OR

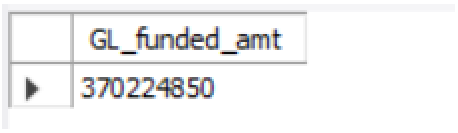
```
SELECT COUNT(id) as  GLApplications FROM financial_loan
WHERE loan_status = "Fully Paid" OR loan_status = "Current"
```



	GLApplications
▶	33243

Calculated the Total Disbursed Amount for Good Loans:

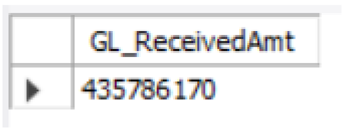
```
SELECT SUM(loan_amount) as  GL_funded_amt FROM financial_loan
WHERE loan_status = "Fully Paid" OR loan_status = "Current";
```



	GL_funded_amt
▶	370224850

Computed the Total Received Amount from Good Loans:

```
SELECT SUM(total_payment) as  GL_ReceivedAmt FROM financial_loan
WHERE loan_status = "Fully Paid" OR loan_status = "Current";
```

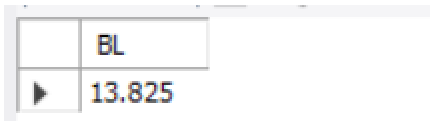


	GL_ReceivedAmt
▶	435786170

Bad Loans: Loans where borrowers fail to repay on time or default entirely; classified as *"Charged Off"* under loan status.

Analyzed the Percentage of Bad Loans (BL):

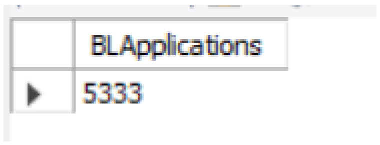
```
SELECT
	ROUND((COUNT(CASE WHEN loan_status = "Charged Off" THEN id END))/COUNT(id)*100,3)
	as BL
FROM financial_loan;
```



BL
13.825

Retrieved the Total Number of Bad Loan Applications:

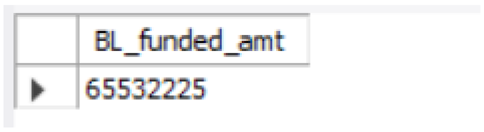
```
SELECT COUNT(id) as  BLApplications FROM financial_loan
WHERE loan_status = "Charged Off";
```



BLApplications
5333

Calculated the Total Disbursed Amount for Bad Loans:

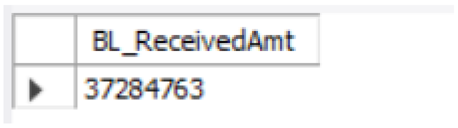
```
SELECT SUM(loan_amount) as  BL_funded_amt FROM financial_loan
WHERE loan_status = "Charged Off";
```



BL_funded_amt
65532225

Computed the Total Received Amount from Bad Loans:

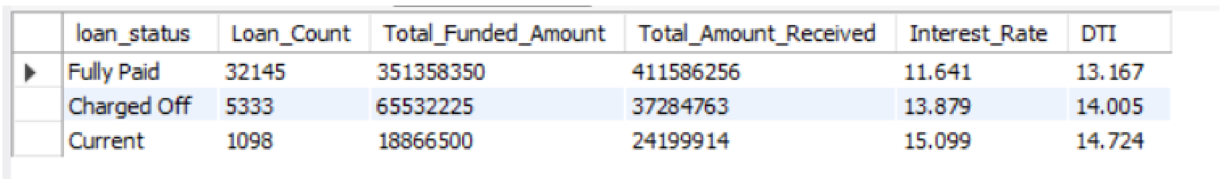
```
SELECT SUM(total_payment) as  BL_ReceivedAmt FROM financial_loan
WHERE loan_status = "Charged Off";
```



BL_ReceivedAmt
37284763

Overview of the Loan Status Grid:

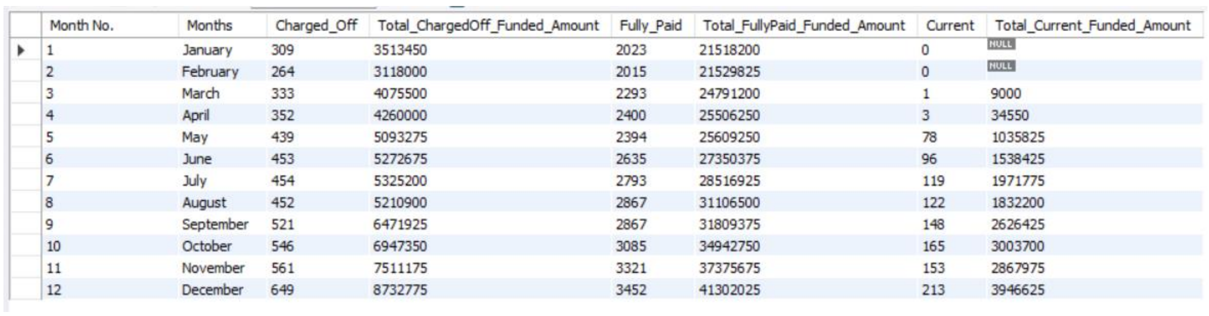
```
SELECT
        loan_status,
        COUNT(id) as Loan_Count,
        SUM(loan_amount) as Total_Funded_Amount,
        SUM(total_payment) as Total_Amount_Received,
        ROUND(AVG(int_rate)*100,3) as Interest_Rate,
        ROUND(AVG(dti)*100,3) as DTI
FROM financial_loan
GROUP BY loan_status;
```



loan_status	Loan_Count	Total_Funded_Amount	Total_Amount_Received	Interest_Rate	DTI
Fully Paid	32145	351358350	411586256	11.641	13.167
Charged Off	5333	65532225	37284763	13.879	14.005
Current	1098	18866500	24199914	15.099	14.724

Analyzed the Monthly Loan Count and Disbursed Amount by Loan Status:

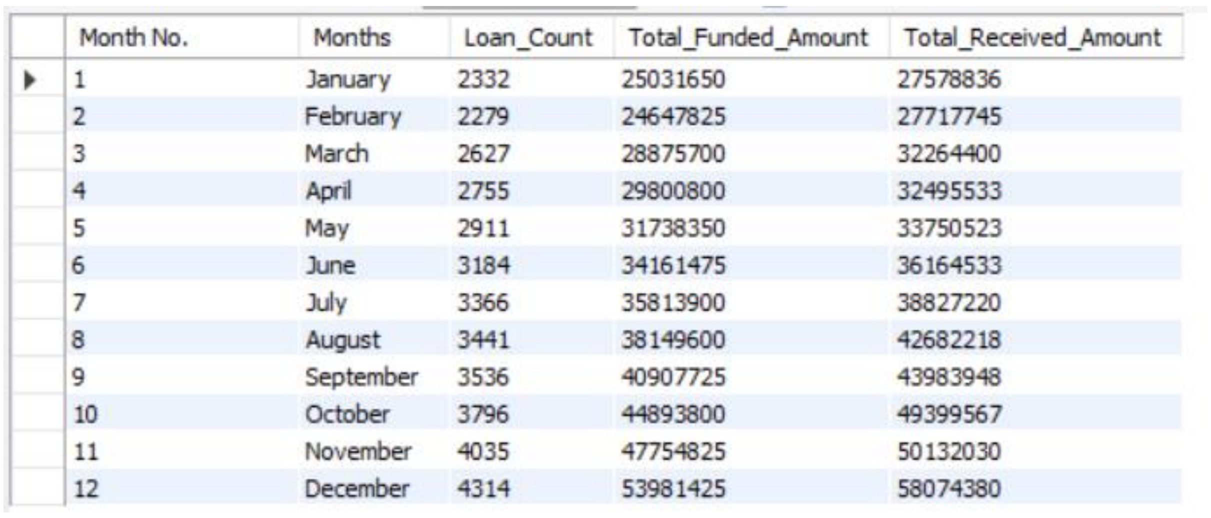
```
SELECT
        MONTH(issue_date) as "Month No.",
        MONTHNAME(issue_date) as "Months",
        COUNT(CASE WHEN loan_status = 'Charged Off' THEN id END) AS Charged_Off,
        SUM(CASE WHEN loan_status = 'Charged Off' THEN loan_amount END) AS
        Total_ChargedOff_Funded_Amount,
        COUNT(CASE WHEN loan_status = 'Fully Paid' THEN id END) AS Fully_Paid,
        SUM(CASE WHEN loan_status = 'Fully Paid' THEN loan_amount END) AS
        Total_FullyPaid_Funded_Amount,
        COUNT(CASE WHEN loan_status = 'Current' THEN id END) AS "Current",
        SUM(CASE WHEN loan_status = 'Current' THEN loan_amount END) AS
        Total_Current_Funded_Amount
FROM financial_loan
GROUP BY month(issue_date), monthname(issue_date)
ORDER BY month(issue_date);
```



Month No.	Months	Charged_Off	Total_ChargedOff_Funded_Amount	Fully_Paid	Total_FullyPaid_Funded_Amount	Current	Total_Current_Funded_Amount
1	January	309	3513450	2023	21518200	0	NULL
2	February	264	3118000	2015	21529825	0	NULL
3	March	333	4075500	2293	24791200	1	9000
4	April	352	4260000	2400	25506250	3	34550
5	May	439	5093275	2394	25609250	78	1035825
6	June	453	5272675	2635	27350375	96	1538425
7	July	454	5325200	2793	28516925	119	1971775
8	August	452	5210900	2867	31106500	122	1832200
9	September	521	6471925	2867	31809375	148	2626425
10	October	546	6947350	3085	34942750	165	3003700
11	November	561	7511175	3321	37375675	153	2867975
12	December	649	8732775	3452	41302025	213	3946625

Analyzed the Monthly Loan Count, Disbursed Amount and Received Amount:

```
SELECT
        MONTH(issue_date) as "Month No.",
        MONTHNAME(issue_date) as "Months",
        COUNT(id) as Loan_Count,
        SUM(loan_amount) as Total_Funded_Amount,
        SUM(total_payment) as Total_Received_Amount
FROM financial_loan
GROUP BY month(issue_date), monthname(issue_date)
ORDER BY month(issue_date);
```

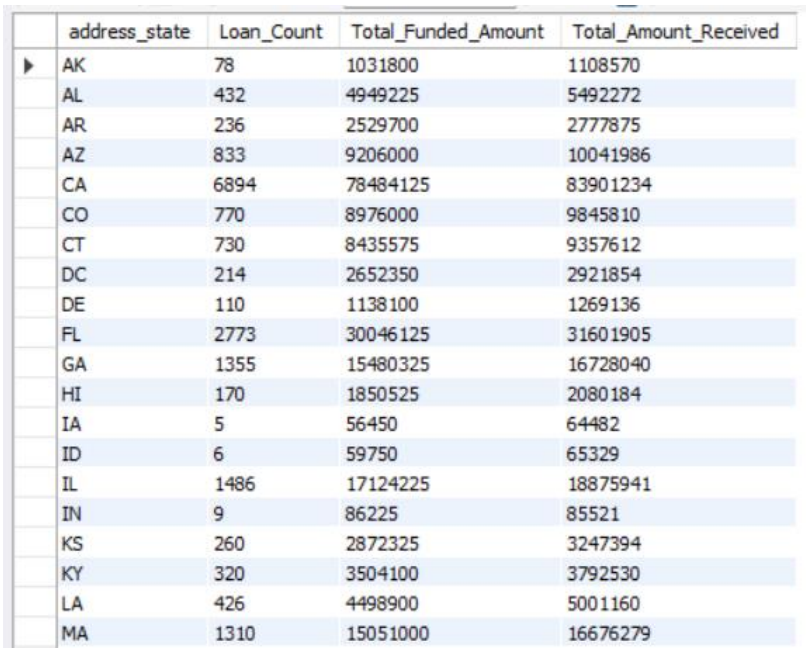


Month No.	Months	Loan_Count	Total_Funded_Amount	Total_Received_Amount
1	January	2332	25031650	27578836
2	February	2279	24647825	27717745
3	March	2627	28875700	32264400
4	April	2755	29800800	32495533
5	May	2911	31738350	33750523
6	June	3184	34161475	36164533
7	July	3366	35813900	38827220
8	August	3441	38149600	42682218
9	September	3536	40907725	43983948
10	October	3796	44893800	49399567
11	November	4035	47754825	50132030
12	December	4314	53981425	58074380

Analyzed Loan Count, Disbursed Amount and Received Amount by Location:

```
SELECT
        address_state,
        COUNT(id) as Loan_Count,
        SUM(loan_amount) as Total_Funded_Amount,
        SUM(total_payment) as Total_Amount_Received
```

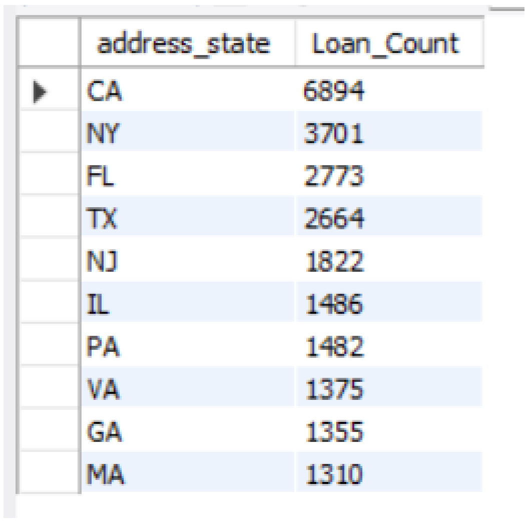
```
FROM financial_loan

GROUP BY address_state

ORDER BY address_state;
```



	address_state	Loan_Count	Total_Funded_Amount	Total_Amount_Received
▶	AK	78	1031800	1108570
	AL	432	4949225	5492272
	AR	236	2529700	2777875
	AZ	833	9206000	10041986
	CA	6894	78484125	83901234
	CO	770	8976000	9845810
	CT	730	8435575	9357612
	DC	214	2652350	2921854
	DE	110	1138100	1269136
	FL	2773	30046125	31601905
	GA	1355	15480325	16728040
	HI	170	1850525	2080184
	IA	5	56450	64482
	ID	6	59750	65329
	IL	1486	17124225	18875941
	IN	9	86225	85521
	KS	260	2872325	3247394
	KY	320	3504100	3792530
	LA	426	4498900	5001160
	MA	1310	15051000	16676279

Identified the Top 10 Locations with Highest Loan Counts:

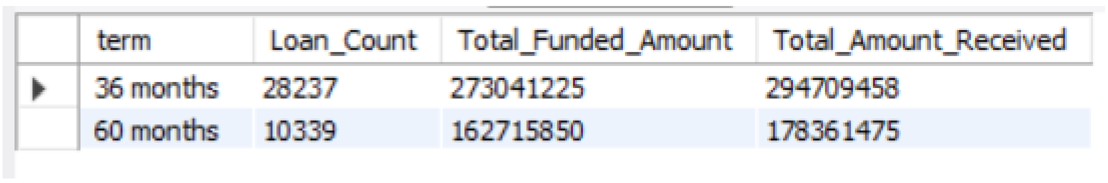
```
SELECT

        address_state,

        COUNT(id) as Loan_Count

FROM financial_loan

GROUP BY address_state

ORDER BY count(id) DESC

LIMIT 10;
```



	address_state	Loan_Count
▶	CA	6894
	NY	3701
	FL	2773
	TX	2664
	NJ	1822
	IL	1486
	PA	1482
	VA	1375
	GA	1355
	MA	1310

Analyzed Loan Count, Disbursed Amount and Received Amount by Term Loan:

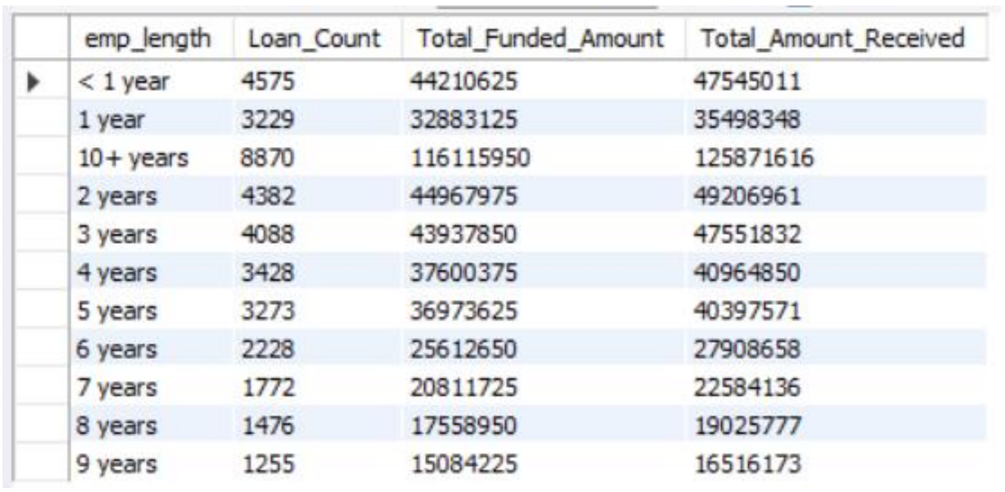
```
SELECT
        term,
        COUNT(id) as Loan_Count,
        SUM(loan_amount) as Total_Funded_Amount,
        SUM(total_payment) as Total_Amount_Received
FROM financial_loan
GROUP BY term
ORDER BY term;
```



term	Loan_Count	Total_Funded_Amount	Total_Amount_Received
36 months	28237	273041225	294709458
60 months	10339	162715850	178361475

Analyzed Loan Count, Disbursed Amount and Received Amount by Employee Length:

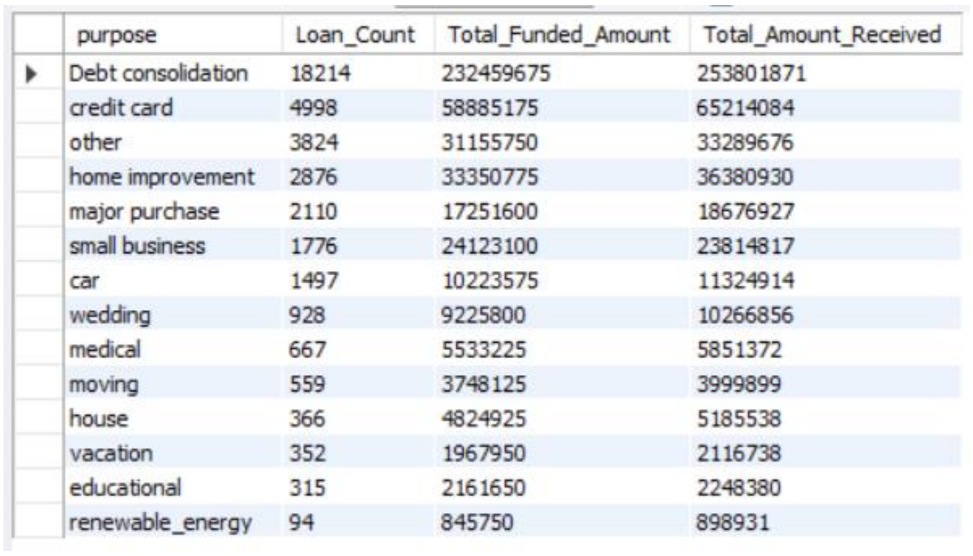
```
SELECT
        emp_length,
        COUNT(id) as Loan_Count,
        SUM(loan_amount) as Total_Funded_Amount,
        SUM(total_payment) as Total_Amount_Received
FROM financial_loan
GROUP BY emp_length
ORDER BY emp_length;
```



emp_length	Loan_Count	Total_Funded_Amount	Total_Amount_Received
< 1 year	4575	44210625	47545011
1 year	3229	32883125	35498348
10+ years	8870	116115950	125871616
2 years	4382	44967975	49206961
3 years	4088	43937850	47551832
4 years	3428	37600375	40964850
5 years	3273	36973625	40397571
6 years	2228	25612650	27908658
7 years	1772	20811725	22584136
8 years	1476	17558950	19025777
9 years	1255	15084225	16516173

Analyzed Loan Count, Disbursed Amount and Received Amount by Loan Purpose:

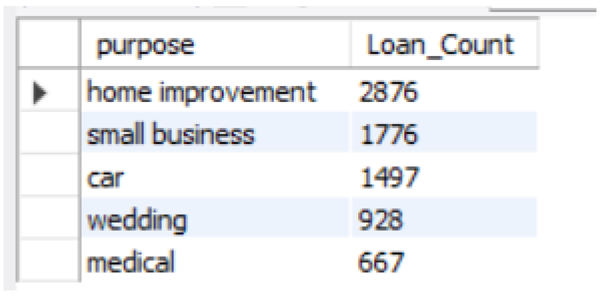
```
SELECT
	purpose,
	COUNT(id) as Loan_Count,
	SUM(loan_amount) as Total_Funded_Amount,
	SUM(total_payment) as Total_Amount_Received
FROM financial_loan
GROUP BY purpose
ORDER BY COUNT(id) DESC;
```



purpose	Loan_Count	Total_Funded_Amount	Total_Amount_Received
Debt consolidation	18214	232459675	253801871
credit card	4998	58885175	65214084
other	3824	31155750	33289676
home improvement	2876	33350775	36380930
major purchase	2110	17251600	18676927
small business	1776	24123100	23814817
car	1497	10223575	11324914
wedding	928	9225800	10266856
medical	667	5533225	5851372
moving	559	3748125	3999899
house	366	4824925	5185538
vacation	352	1967950	2116738
educational	315	2161650	2248380
renewable_energy	94	845750	898931

Identified the Top 5 Loan Purposes with the Highest Loan Counts, Excluding Non-Specific Categories:

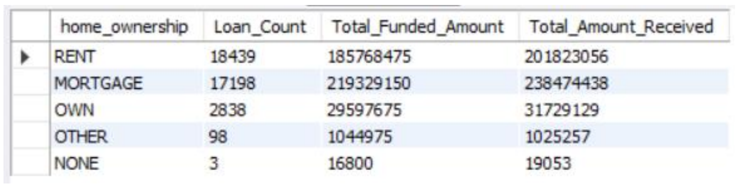
```
SELECT
	purpose,
	COUNT(id) as Loan_Count
FROM financial_loan
WHERE purpose NOT IN ("credit card", "Debt consolidation", "other", "major purchase")
GROUP BY purpose
ORDER BY COUNT(id) DESC
LIMIT 5;
```



purpose	Loan_Count
home improvement	2876
small business	1776
car	1497
wedding	928
medical	667

Analyzed Loan Count, Disbursed Amount and Received Amount by Home Ownership:

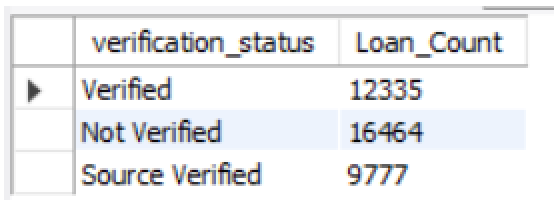
```
SELECT
        home_ownership,
        COUNT(id) as Loan_Count,
        SUM(loan_amount) as Total_Funded_Amount,
        SUM(total_payment) as Total_Amount_Received
FROM financial_loan
GROUP BY home_ownership
ORDER BY COUNT(id) DESC;
```



home_ownership	Loan_Count	Total_Funded_Amount	Total_Amount_Received
RENT	18439	185768475	201823056
MORTGAGE	17198	219329150	238474438
OWN	2838	29597675	31729129
OTHER	98	1044975	1025257
NONE	3	16800	19053

Identified the Verification Status of Loan Applications:

```
SELECT
        verification_status,
        COUNT(id) as Loan_Count
FROM financial_loan
GROUP BY verification_status;
```



verification_status	Loan_Count
Verified	12335
Not Verified	16464
Source Verified	9777