Uncovering Financial Insights with SQL

Description: This SQL project explores financial data through advanced querying techniques to derive valuable insights from loan and payment datasets. By combining data from two datasets, the analysis provides a comprehensive view of financial health, empowering decision-makers to manage risk and improve financial strategies. The queries and insights are designed to demonstrate practical SQL skills applicable to financial analysis, making this project a valuable portfolio addition for aspiring data analysts and financial professionals.

MySQL

XLS

Key Insights:

1. Loan distribution by purpose and state highlights areas of high financial activity.
2. Default rates by loan grade provide insights into risk assessment.
3. Delinquency analysis helps identify borrowers likely to default.
4. Payment recovery patterns shed light on financial recovery efforts.
5. Correlating debt-to-income (DTI) with open accounts reveals borrower risk profiles.

Tools: MySQL ,MS-Excel

Total Counts of rows

```
6   -- Total count of row
7 • select count(*) from finance_1;
8
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(*)
39717

```
11
12 • select count(*) from finance_2;
13
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

count(*)
39717

```sql
-- Question 1 : Calculate Total Loan Amount by Purpose

select purpose ,sum(loan_amnt) as total_loan_amnt from finance_1
group by purpose
order by total_loan_amnt desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

purpose	total_loan_amnt
debt_consolidation	236647300
credit_card	60142150
home_improvement	34334725
other	32213975
small_business	24800975
major_purchase	17835925
car	10498375
wedding	9352600
medical	5726725

```
24   -- Question 2: Find the Average Interest Rate by Grade
25
26 • select grade, round(avg(int_rate),2) as avg_int_rate from finance_1
27   group by grade
28   order by grade;
29
30
31
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

grade	avg_int_rate
A	7.34
B	11.02
C	13.55
D	15.72
E	17.71
F	19.75
G	21.4

```
2   -- Question 3: Identify the Top 5 States with the Highest Loan Amounts
3
4 • select addr_state,loan_amnt from finance_1 order by addr_state desc limit 5;
5
6
7
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

addr_state	loan_amnt
WY	13000
WY	27575
WY	25000
WY	16000
WY	25000

```sql
-- Question 4: Calculate the Default Rate by Grade
SELECT grade,COUNT(CASE WHEN loan_status = 'Charged Off' THEN 1 END) * 100.0 / COUNT(*) AS default_rate
FROM finance_1
GROUP BY grade
ORDER BY default_rate DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

grade	default_rate
G	31.96203
F	30.40991
E	25.15834
D	21.06652
C	16.63374
B	11.85524
A	5.96926

```
46    -- Question 5: Find Borrowers with More than 10 Open Accounts
47
48 •  select id,open_acc from finance_2
49    where open_acc > 10;
50
51
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

id	open_acc
57416	14
59006	13
61390	11
65426	14
65640	20
66964	11
68163	14
69251	16
69924	11

```
2    -- Question 6: Calculate Total Payments and Recoveries
3 •  select round(sum(total_pymnt),2) as total_pymnt ,
4           round(sum(recoveries),2) as total_recoveries from finance_2;
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

total_pymnt	total_recoveries
482704393.92	3781937


```sql
-- Question 7: Calculate Debt-to-Income Ratio (DTI) by State
select addr_state , round(avg(dti),2) as Debt_to_Income from finance_1
group by addr_state
order by Debt_to_Income desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

addr_state	Debt_to_Income
NE	16.28
IN	15.97
AR	15.3
WV	15.08
MS	15
AK	14.86
SD	14.75
ID	14.54
OH	14.52

```
67    -- Question 8: List Borrowers with the Highest Installment Amounts
68
69 •  SELECT member_id, loan_amnt, installment
70    FROM finance_1
71    ORDER BY installment DESC
72    LIMIT 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

member_id	loan_amnt	installment
1008051	35000	1305.19
934508	35000	1302.69
939091	35000	1295.21
999775	35000	1288.1
1086945	35000	1288.1
1268129	35000	1283.5
1157666	35000	1276.6
1254299	35000	1276.6
1094680	35000	1276.6


```sql
-- Question 9: Join Tables to Find Total Payments by Loan Purpose

SELECT f1.purpose, ROUND(SUM(f2.total_pymnt), 2) AS total_payments
FROM finance_1 f1
JOIN finance_2 f2 ON f1.member_id = f2.id
GROUP BY f1.purpose
ORDER BY total_payments DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

purpose	total_payments
debt_consolidation	9842041.53
credit_card	2179893.63
other	2161594.25
home_improvement	1409175.1
small_business	1242538.17
major_purchase	1046435.51
car	818583.02
wedding	581616.31
moving	493033.14

```sql
-- Question 10: Find Total Recoveries by Loan Grade
select finance_1.grade ,sum( finance_2.recoveries) as total_recoveries from finance_1
join finance_2
on finance_1.member_id = finance_2.id
group by grade;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

grade	total_recoveries
C	45559
A	32652
D	6804
B	64377
E	9746
F	1515
G	17275

```sql
-- Question 11 : Grade -Subcategory wise revolution balance

select finance_1.grade,finance_1.sub_grade, sum(finance_2.revol_bal) as total_revol_bal
from finance_1
inner join finance_2
on finance_1.member_id = finance_2.id
group by grade,sub_grade
order by grade,sub_grade;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

grade	sub_grade	total_revol_bal
A	A1	487627
A	A2	752698
A	A3	1182653
A	A4	1688063
A	A5	1772486
B	B1	775384
B	B2	1083485

```sql
-- Question 12 total paymne_gradet for verified status Vs Non verified status

select finance_1.verification_status ,round(sum(finance_2.total_pymnt),2) as total_pymnt
from finance_1
inner join finance_2
on finance_1.member_id = finance_2.id
group by verification_status;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

verification_status	total_pymnt
Not Verified	9165365.84
Verified	6976427.78
Source Verified	4781492.28

THANK YOU