



LENDING CLUB PROJECT

SQL Coding

Author: Syafiq Azman

CONTACT: 0478664878

EMAIL: syafiqazmn@gmail.com



Table of Contents

Section #1: Server Connection and Simple SQL Queries (2 Marks)	3
Part 1: Connection to SQL Server on the Cloud (1 Mark)	3
Part 2: Simple SELECT Query (1 Mark)	3
Section #2: SQL for Business Insights (20 marks).....	4
Part 1: Loan and Funded Amounts (3 marks).....	4
Part 2: Loan Terms (2 Marks).....	5
Part 3: Interest Rate (3 Marks)	6
Part 4: Loan Status (2 Marks).....	7
Part 5: Loan Grades (5 Marks)	13
Part 6: Loan Defaults/Delinquencies (5 Marks)	14
Section #3: SQL and Data Visualisation for Business Insights (13 Marks).....	15
Part 1: Data Acquisition and Data Visualisations	15

Section #1: Server Connection and Simple SQL Queries (2 Marks)

Part 1: Connection to SQL Server on the Cloud (1 Mark)

```
--creating table
create table lendingclub (
  id integer primary key,
  member_id integer,
  loan_amnt real,
  funded_amnt real,
  term text,
  int_rate real,
  installment real,
  grade text,
  sub_grade text,
  emp_length text,
  home_ownership text,
  annual_inc real,
  verification_status text,
  issue_d text,
  loan_status text,
  description text,
  purpose text,
  addr_state text,
  dti real,
  earliest_cr_line text,
  out_prncp real,
  total_pymnt real,
  last_pymnt_d text,
  last_pymnt_amnt real,
  application_type text,
  recoveries real
);
```

Part 2: Simple SELECT Query (1 Mark)

1. Create a SELECT statement that selects the first 100 rows for all columns of data.

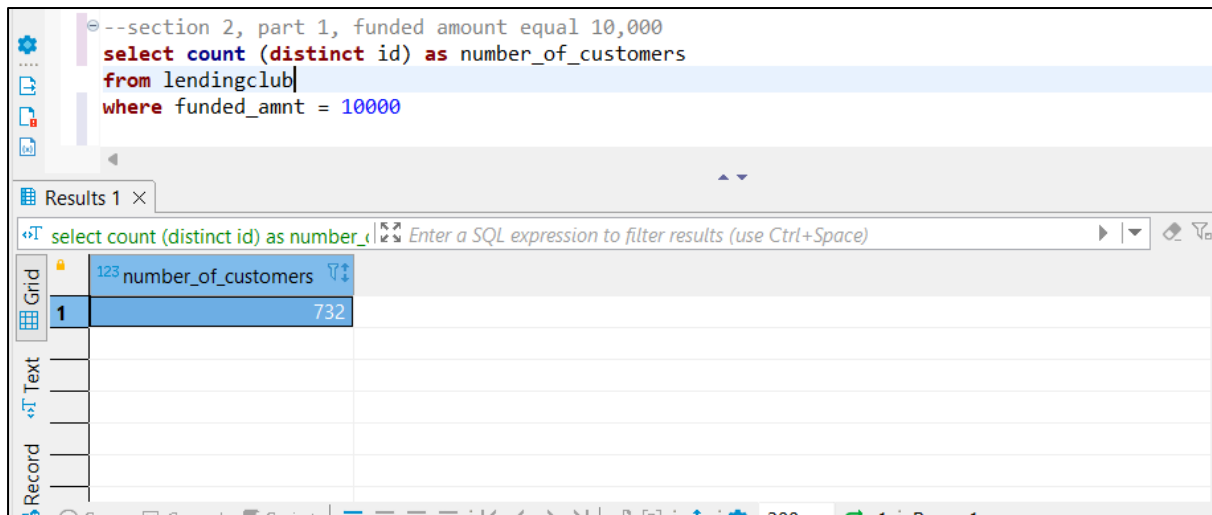
```
--section 1, part 2, first 100 rows for all columns
select * from lendingclub
limit 100
```

	id	member_id	loan_amnt	funded_amnt	term	int_rate	installment	Value
97	1,338,085	1,583,248	20,000	20,000	60 months	21.479999542	546.489990234	E
98	38,321,912	41,105,662	3,000	3,000	36 months	12.390000343	100.209999084	C
99	39,641,917	42,465,760	11,000	11,000	36 months	13.659999847	374.149993896	C
100	41,369,443	44,276,224	28,000	28,000	60 months	10.989999771	608.650024414	B

Section #2: SQL for Business Insights (20 marks)

Part 1: Loan and Funded Amounts (3 marks)

- Execute the SQL script provide for loan amounts. Interpret the results of the query.
- Modify the SQL statement to show the number customers where the funded amount is:
 - o Equal to \$10,000 dollars



```
--section 2, part 1, funded amount equal 10,000
select count (distinct id) as number_of_customers
from lendingclub
where funded_amnt = 10000
```

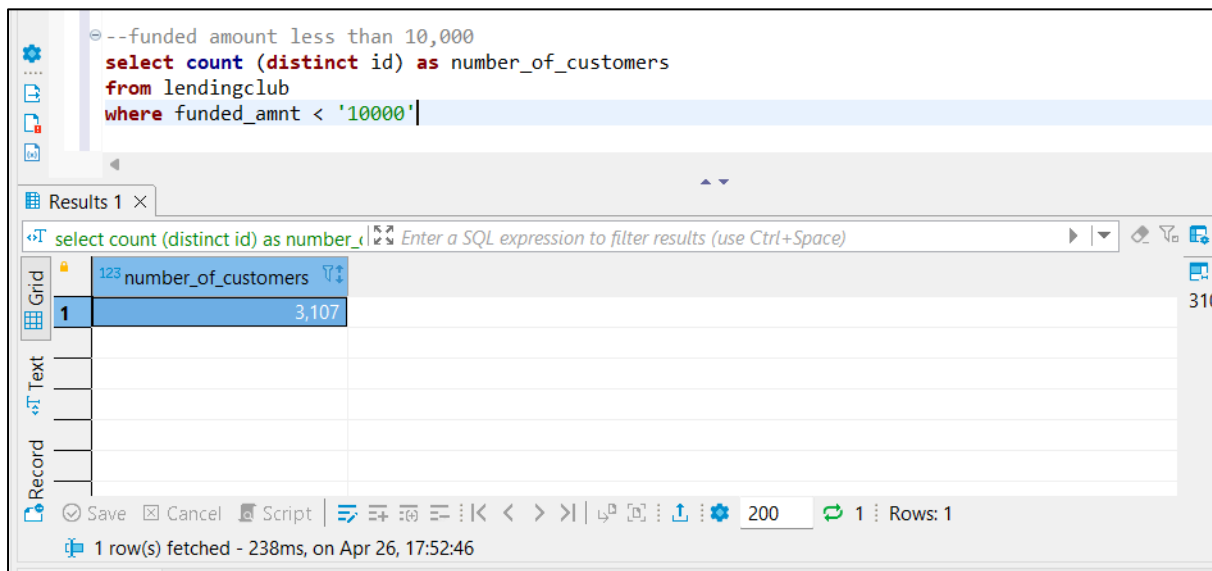
Results 1 x

select count (distinct id) as number_of_customers

	number_of_customers
1	732

There are 732 customers with funded amount equal to \$10,000

- o Less than \$10,000 dollars



```
--funded amount less than 10,000
select count (distinct id) as number_of_customers
from lendingclub
where funded_amnt < '10000'
```

Results 1 x

select count (distinct id) as number_of_customers

	number_of_customers
1	3,107

1 row(s) fetched - 238ms, on Apr 26, 17:52:46

There are 3107 customers with funded amount less than \$10,000

- o Greater than \$10,000 dollars

```
--funded amount greater than 10,000
select count (distinct id) as number_of_customers
from lendingclub
where funded_amnt > '10000'
```

Results 1 x

select count (distinct id) as number_of_customers

	number_of_customers
1	6,161

1 row(s) fetched - 346ms, on Apr 26, 17:53:18

There are 6161 customers with funded amount greater than \$10,000

Part 2: Loan Terms (2 Marks)

Create an SQL statement that counts the terms of the loans. Take a screenshot of your SQL

statement and the corresponding results and paste these into your report.

- How many terms are of 36 months? How many are of 60 months?

```
--section 2, part 2, how many terms for 36 and 60 months
select count (distinct member_id) as number_of_customers, term as term_duration
from lendingclub
group by term
```

lendingclub 1 x

select count (distinct member_id) as number_of_customers, term as term_duration

	number_of_customers	term_duration
1	7,051	36 months
2	2,949	60 months

2 row(s) fetched - 182ms, on Apr. 28, 14:50:04

There are 7051 customers with 36 months term and 2949 customer with 60 months term.

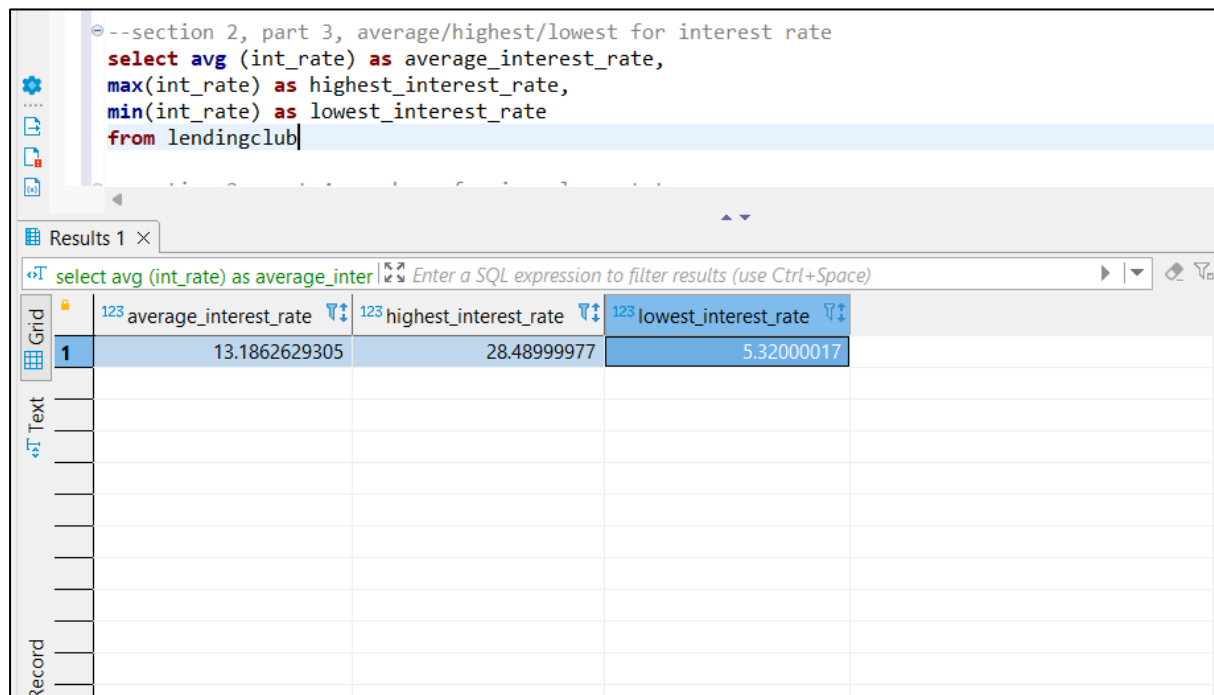
Part 3: Interest Rate (3 Marks)

Create an SQL statement that answers the following business questions:

- What is the average interest rate?
- What is the highest interest rate?
- What is the lowest interest rate?

Take a screenshot of your SQL statement and the corresponding results and paste these into your

report.



```
--section 2, part 3, average/highest/lowest for interest rate
select avg (int_rate) as average_interest_rate,
max(int_rate) as highest_interest_rate,
min(int_rate) as lowest_interest_rate
from lendingclub
```

	123 average_interest_rate	123 highest_interest_rate	123 lowest_interest_rate
1	13.1862629305	28.48999977	5.32000017

The average interest rate is 13.19, the highest interest rate is 28.49 and the lowest interest rate is 5.32

Part 4: Loan Status (2 Marks)

Create an SQL statement that:

- Counts the number of unique loan statuses.

Unique loan status

The screenshot shows a SQL IDE with a query editor and a results grid. The query editor contains the following SQL statement:

```
--section 2, part 4, number of unique loan statuses
select distinct loan_status as unique_loan_status
from lendingclub
order by loan_status asc;
```

The results grid displays the following data:

	unique_loan_status
1	Charged Off
2	Current
3	Default
4	Does not meet the credit policy. Status:Charged Off
5	Does not meet the credit policy. Status:Fully Paid
6	Fully Paid
7	In Grace Period
8	Issued
9	Late (16-30 days)
10	Late (31-120 days)

Count of unique loan status

The screenshot shows a SQL IDE with a query editor and a results grid. The query editor contains the following SQL statement:

```
--count of unique loan status
select count (distinct loan_status) as unique_loan_status
from lendingclub
```

The results grid displays the following data:

	unique_loan_status
1	10

• For each of the unique statuses found above, create an SQL statement that displays the first

100 rows of data.

For Charged Off

The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
--For each of the unique statuses found above, create an SQL statement that displays the first 100 rows
--for charged off
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Charged Off'
limit 100
```

The results grid displays 100 rows of data for the 'Charged Off' status. The columns are 'id' (labeled as 'number_of_customers') and 'loan_status'. The status is consistently 'Charged Off' for all rows. The 'id' values range from 4,282,031 to 524,194. A 'Value' pane on the right shows the value 2824826.

	number_of_customers	loan_status
90	4,282,031	Charged Off
91	4,785,402	Charged Off
92	3,932,991	Charged Off
93	46,794,818	Charged Off
94	1,605,585	Charged Off
95	1,155,362	Charged Off
96	516,726	Charged Off
97	8,576,455	Charged Off
98	20,137,543	Charged Off
99	35,094,088	Charged Off
100	524,194	Charged Off

For Current

The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
--for current
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Current'
limit 100
```

The results grid displays 100 rows of data for the 'Current' status. The columns are 'id' (labeled as 'number_of_customers') and 'loan_status'. The status is consistently 'Current' for all rows. The 'id' values range from 66,570,626 to 49,157,179. A status bar at the bottom indicates '100 row(s) fetched: 210ms on Apr 26, 19:46:27'.

	number_of_customers	loan_status
90	66,570,626	Current
91	15,400,607	Current
92	65,686,982	Current
93	20,670,421	Current
94	16,362,043	Current
95	54,101,073	Current
96	58,001,134	Current
97	59,130,678	Current
98	57,325,521	Current
99	35,024,801	Current
100	49,157,179	Current

For Default

The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
--for default
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Default'
limit 100
```

The results grid displays the following data:

	number_of_customers	loan_status
8	34,934,596	Default
9	41,257,304	Default
10	8,965,283	Default
11	40,796,576	Default

For Does not meet the credit policy. Status Charged Off

The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
--for Does not meet the credit policy. Status:Charged Off
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Does not meet the credit policy. Status:Charged Off'
limit 100
```

The results grid displays the following data:

	number_of_customers	loan_status
5	352,207	Does not meet the credit policy. Status:Charged Off
6	267,470	Does not meet the credit policy. Status:Charged Off
7	325,121	Does not meet the credit policy. Status:Charged Off
8	419,024	Does not meet the credit policy. Status:Charged Off

At the bottom of the IDE, a status bar indicates: "8 row(s) fetched - 121ms, on Apr. 28, 15:31:40".

For Does not meet the credit policy. Status: Fully Paid

The screenshot shows a SQL IDE interface. The top pane contains a SQL query:

```
--for Does not meet the credit policy. Status:Fully Paid
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Does not meet the credit policy. Status:Fully Paid'
limit 100
```

The bottom pane shows the query results in a grid view. The first column is labeled 'number_of_customers' and the second is 'loan_status'. The results are as follows:

	number_of_customers	loan_status
16	586,998	Does not meet the credit policy. Status:Fully Paid
17	69,285	Does not meet the credit policy. Status:Fully Paid
18	493,840	Does not meet the credit policy. Status:Fully Paid
19	557,484	Does not meet the credit policy. Status:Fully Paid

The status bar at the bottom indicates '19 row(s) fetched - 130ms, on Apr. 28, 15:30:56'.

For In Grace Period

The screenshot shows a SQL IDE interface. The top pane contains a SQL query:

```
--for in grace period
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'In Grace Period'
limit 100
```

The bottom pane shows the query results in a grid view. The first column is labeled 'number_of_customers' and the second is 'loan_status'. The results are as follows:

	number_of_customers	loan_status
64	58,250,657	In Grace Period
65	9,196,379	In Grace Period
66	33,161,364	In Grace Period
67	7,086,321	In Grace Period

The status bar at the bottom indicates '67 row(s) fetched - 127ms, on Apr. 28, 15:19:48'.

For Fully Paid

The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
--for fully paid
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Fully Paid'
limit 100
```

The results grid displays 100 rows of data. The first four rows are visible:

	number_of_customers	loan_status
97	5,038,660	Fully Paid
98	27,610,461	Fully Paid
99	10,587,321	Fully Paid
100	24,796,751	Fully Paid

The status bar at the bottom indicates "100 row(s) fetched - 181ms, on Apr. 28, 15:18:46".

For Issued

The screenshot shows a SQL IDE with a query editor at the top and a results grid below. The query is:

```
--for issued
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Issued'
limit 100
```

The results grid displays 83 rows of data. The first four rows are visible:

	number_of_customers	loan_status
80	67,398,879	Issued
81	68,537,494	Issued
82	68,585,518	Issued
83	68,543,346	Issued

The status bar at the bottom indicates "83 row(s) fetched - 251ms, on Apr. 28, 15:20:25".

For Late (16-30 days)

```
--for late (16-30 days)
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Late (16-30 days)'
limit 100
```

lendingclub 1 ×

select distinct id as number_of_cust... Enter a SQL expression to filter results (use Ctrl+Space)

	number_of_customers	loan_status
20	4,528,689	Late (16-30 days)
21	30,305,773	Late (16-30 days)
22	50,607,566	Late (16-30 days)
23	5,825,820	Late (16-30 days)

Save Cancel Script | 200 23 Rows: 1

23 row(s) fetched - 159ms, on Apr. 28, 15:21:03

For Late (31-120 days)

```
--for late (31-120 days)
select distinct id as number_of_customers, loan_status
from lendingclub
where loan_status = 'Late (31-120 days)'
limit 100
```

lendingclub 1 ×

select distinct id as number_of_cust... Enter a SQL expression to filter results (use Ctrl+Space)

	number_of_customers	loan_status
97	39,349,716	Late (31-120 days)
98	50,083,849	Late (31-120 days)
99	58,612,145	Late (31-120 days)
100	20,070,417	Late (31-120 days)

Save Cancel Script | 200 100 Rows: 1

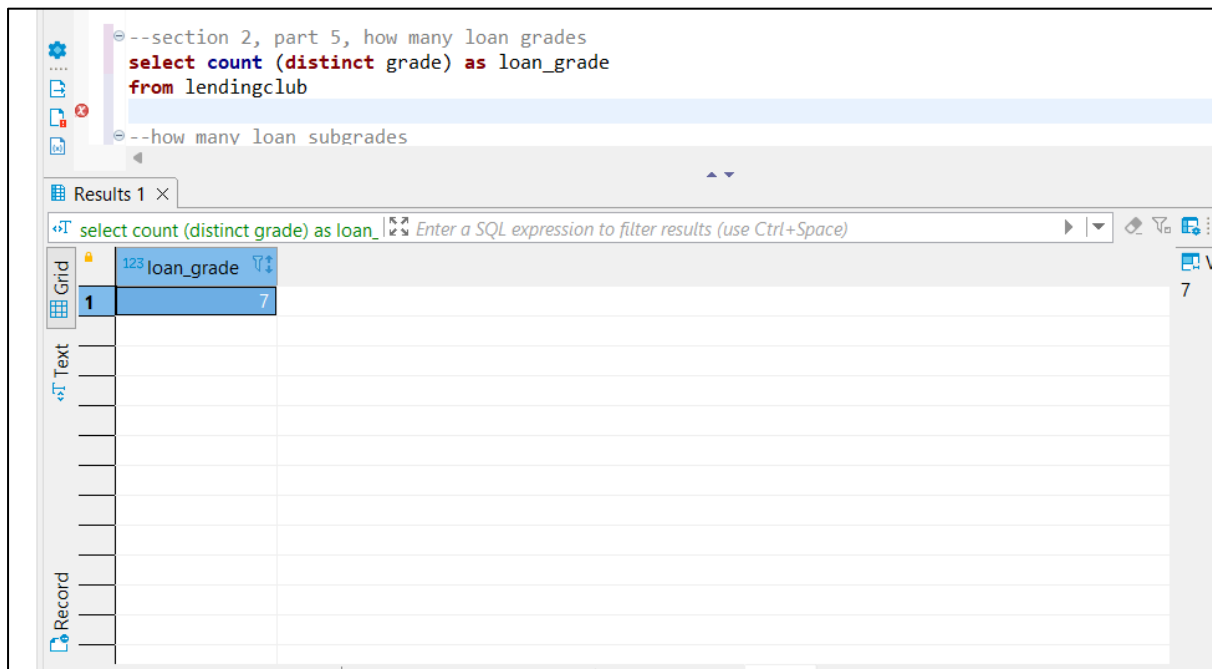
100 row(s) fetched - 632ms, on Apr. 28, 15:21:32

Problems ×

Part 5: Loan Grades (5 Marks)

Create an SQL statement that answers the following business questions:

- How many loan grades are there?



```
--section 2, part 5, how many loan grades
select count (distinct grade) as loan_grade
from lendingclub

--how many loan subgrades
```

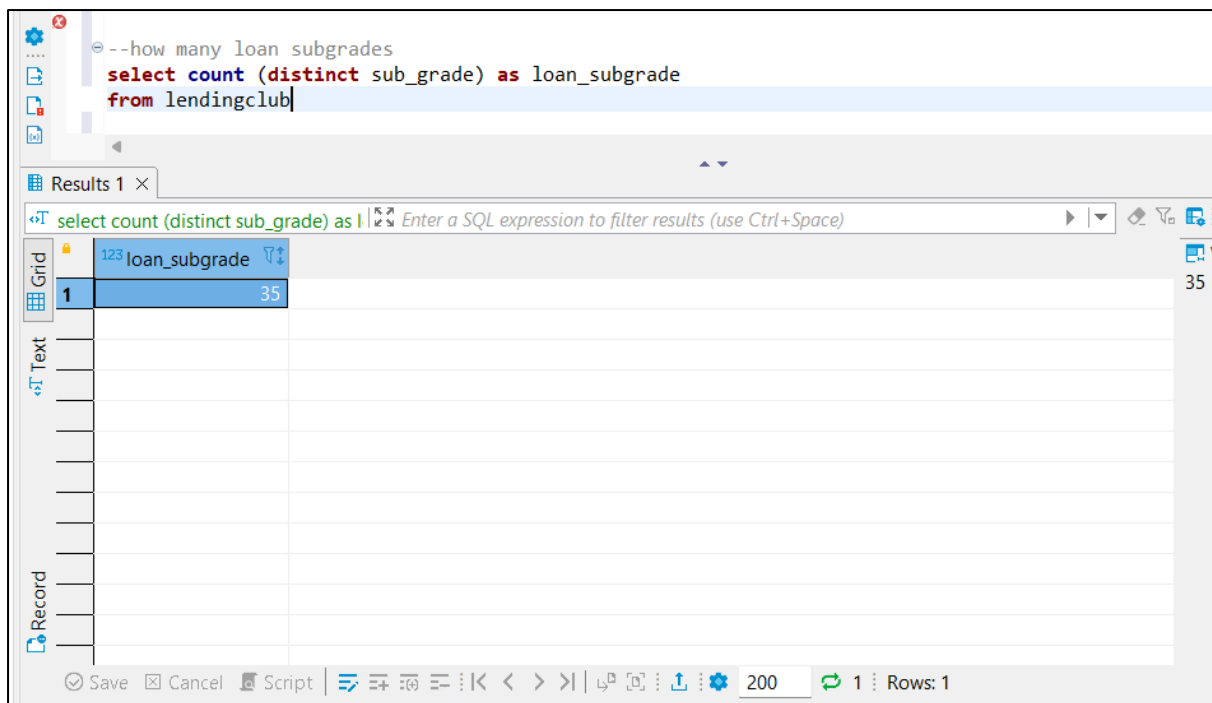
Results 1 x

select count (distinct grade) as loan_grade Enter a SQL expression to filter results (use Ctrl+Space)

	123 loan_grade
1	7

There are 7 loan grades.

- How many loans sub-grades are there?



```
--how many loan subgrades
select count (distinct sub_grade) as loan_subgrade
from lendingclub
```

Results 1 x

select count (distinct sub_grade) as loan_subgrade Enter a SQL expression to filter results (use Ctrl+Space)

	123 loan_subgrade
1	35

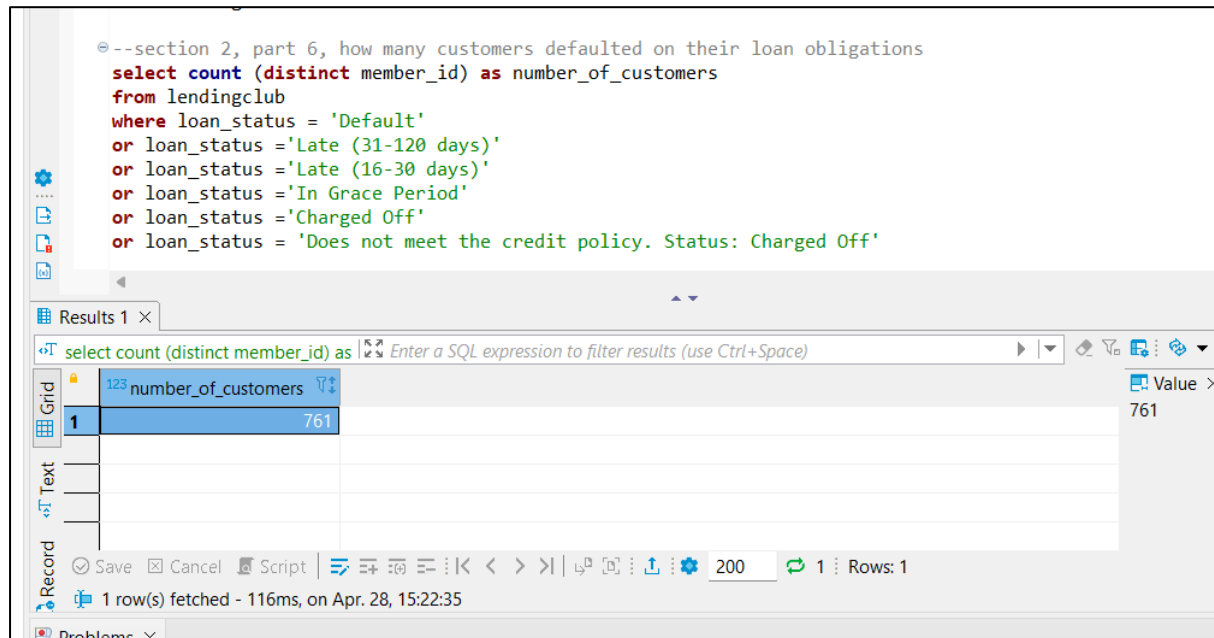
Save Cancel Script 200 1 Rows: 1

There are 35 loan sub-grades.

Part 6: Loan Defaults/Delinquencies (5 Marks)

Create an SQL statement that answers the following questions:

- How many customers defaulted on their loan obligations?



The screenshot shows a SQL IDE interface. The top pane contains a SQL query: `--section 2, part 6, how many customers defaulted on their loan obligations`
`select count (distinct member_id) as number_of_customers`
`from lendingclub`
`where loan_status = 'Default'`
`or loan_status = 'Late (31-120 days)'`
`or loan_status = 'Late (16-30 days)'`
`or loan_status = 'In Grace Period'`
`or loan_status = 'Charged Off'`
`or loan_status = 'Does not meet the credit policy. Status: Charged Off'`

The bottom pane shows the results of the query. The first row is highlighted, showing a count of 761. The column is labeled 'number_of_customers'.

number_of_customers
761

The status bar at the bottom indicates '1 row(s) fetched - 116ms, on Apr. 28, 15:22:35'.

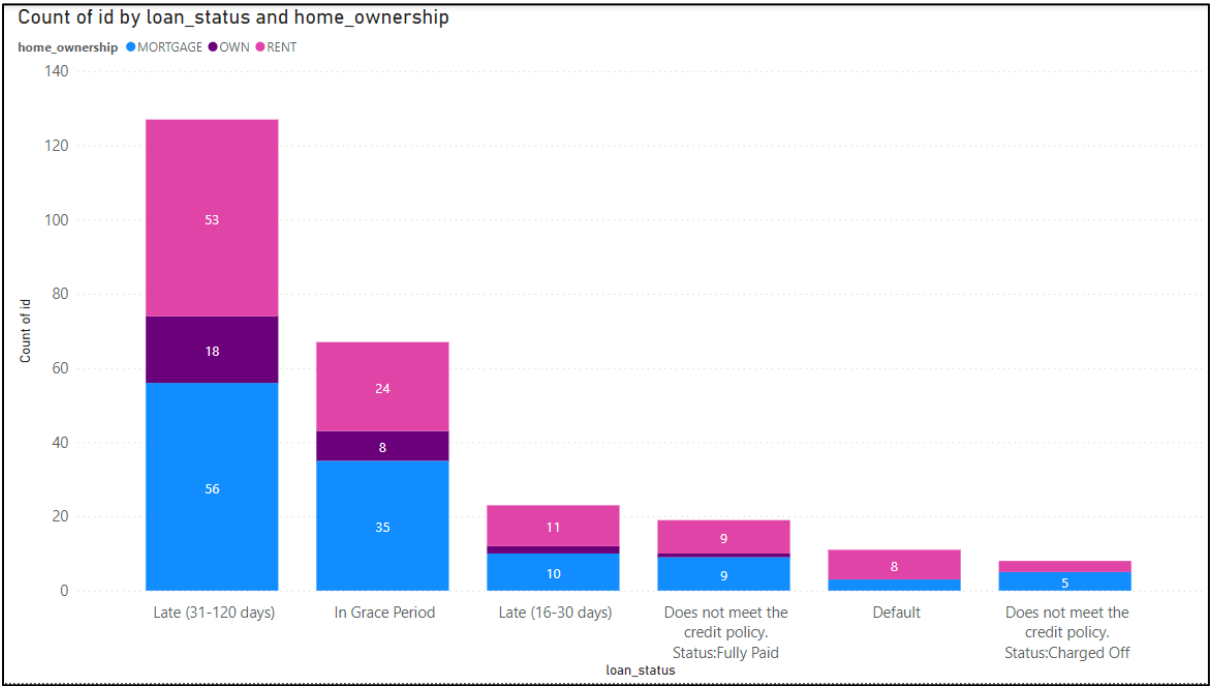
For this case, customers who defaulted refer to customers with loan status default, late (31-120days), late (16-30 days), in grace period, charged off and do not meet the credit policy charged off. 761 customers defaulted on their loans.

Section #3: SQL and Data Visualisation for Business Insights (13 Marks)

Part 1: Data Acquisition and Data Visualisations

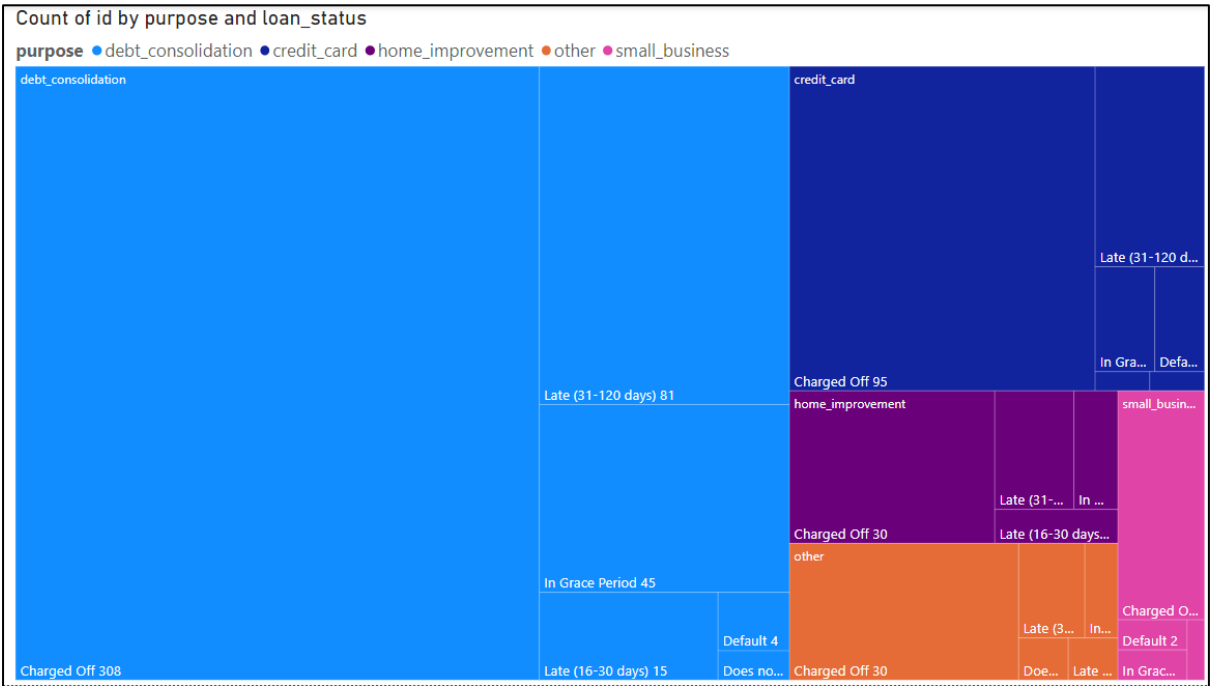
- Create a SELECT statement that returns all rows and columns of data.
- Export and save the results of the query as CSV file.
- Import the CSV file in either PowerBI or Tableau.
- Select the following variables for visualisations:
 - o ID, Member ID, Loan Amount, Funded Amount, Term, Interest Rate, Term, Instalment, Grade, Sub-Grade, Employee Length, Home Ownership, Annual Income, Verification Status, Issue Date, Loan Status, Description, Purpose, Address State, DTI, Earliest Credit Line, Outstanding Principal, Total Payment, Recoveries, Collections, Last Payment Date, Last Payment Amount, and Application Type.
- From the data selected, create 3 kinds of unique visualisations, discover more business insights, and interpret the results. Can you identify the kinds of customers that default on their loans?

Visualisation 1: Customers by Loan Status (Defaulted) and Home Ownership



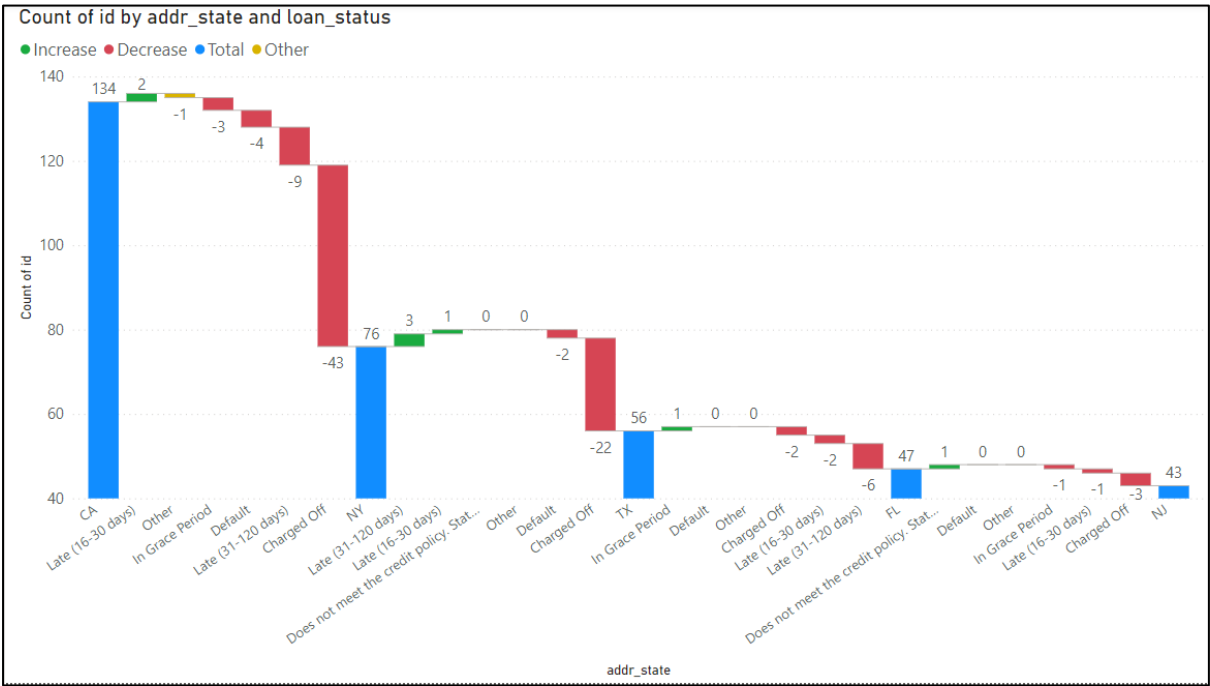
Based on Visualisation 1, we can infer that most customers who defaulted are customers with homeownership, either renting or mortgage, while customers who own a home defaulted the least.

Visualisation 2: Customers by Top 5 Highest Loan Purpose and Loan Status (Defaulted)



Visualisation 2 shows the top 5 purposes where customers who applied them defaulted. A large proportion of customers that defaulted are loaning with the purpose of debt consolidation. The second is for a credit card, followed by home improvement, other and small business.

Visualisation 3: Customers by State and Loan Status (Defaulted)



Visualisation 3 illustrates the top 5 states with the highest number of customers defaulted. First, with 134 customers, California, followed by New York, Texas, Florida and New Jersey with 76, 56, 47 and 43 customers, respectively.