Topic: **Credit Loan Analysis**

Ram Bhikhalal Vaghani: 002704237

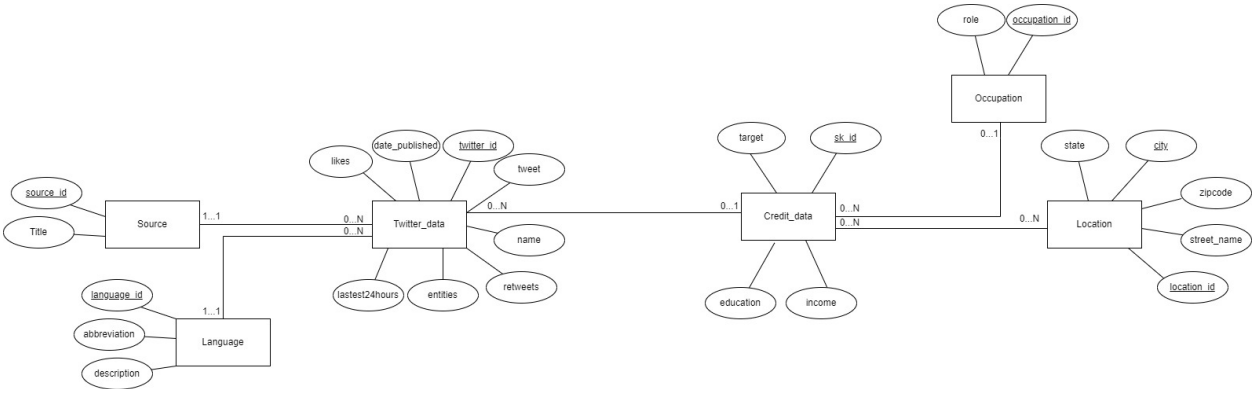
Problem Statement:

The aim to predict whether the client will be able to pay the loan after some X days based on some features like they employed, if they own home, occupation they belong to, their annual income, etc (more details are given in the table below).

Data Sources:

For this analysis, we have scrapped a twitter data. In this twitter data was formatted where the **language_id** from the table **language** and **source_id** from the table **source** act as **foreign key** to the main table **twitter_data**.
In the other table, credit loan data, customer's details asking for the loan is mentioned. This table to connected to the properties they own and the property location detail. These customers could have also tweeted regarding the credit loan. These customers are working in some occupation, the **occupation_id** from the **occupation** table act as the **foreign key** to main **credit** table.

ER Diagram:



Relational Mapping:(Underline – primary key, # - foreign key)

In the relational mapping because of the many-to-many relational sharing between creditloan and location a new relation called 'property' is created.

twitter_data (twitter_id, date_published, likes, tweet, name, retweets, entities, lastest24hours, #sk_id, #source_id, #language_id)
Not Null: source_id, language_id

source (source_id, title)

language (language_id, abbreviation, description)

credit_data (sk_id, target, education, income, #occupation_id)
Not Null: NA

location (location_id, street_name, city, state, zipcode)

occupation (occupation_id, role)

property (#sk_id, #location_id)
Not Null: NA

Data Accuracy/Validity:

Based on entities mapping the ERD all the validations have been validated while creating all the tables. Since the customers are only from USA their property evaluation is also suppose to be from USA.
The following is the completeness and uniformity check that has been performed:
- The data is restricted to customers only from USA
- Every twitter data should have one and only one source and language
- A customer may have writer any number of tweets
- A customer may have any number of homes
- A customer can belong to maximum occupation

Data Completeness/Uniformity:

The data does not contain null values. The null values are only in the foreign key concept which is based on cardinalities.
The following can be validated from the python snippets:
Table customer:

```
custeda.isnull().sum()
```

```
sk_id              0
target             0
income             0
education          0
occupation_id    115
dtype: int64
```

Table Occupation:

```
empeda.isnull().sum()
```

```
occupation_id    0
role             0
dtype: int64
```

Table source:

```
source.isnull().sum()
```

```
source_id    0
title        0
dtype: int64
```

Table Twitter_data:

```
twitter_data.isnull().sum()
```

```
Unnamed: 0          0
Twitter_Id          0
Date_Published      0
Lik210              0
source_id           0
Tweet               0
Name                0
language_id         0
 Retweets           0
200titi210          0
Lat210t24hours    170
sk_id              59
dtype: int64
```

Table Location:

```
location.isnull().sum()
```

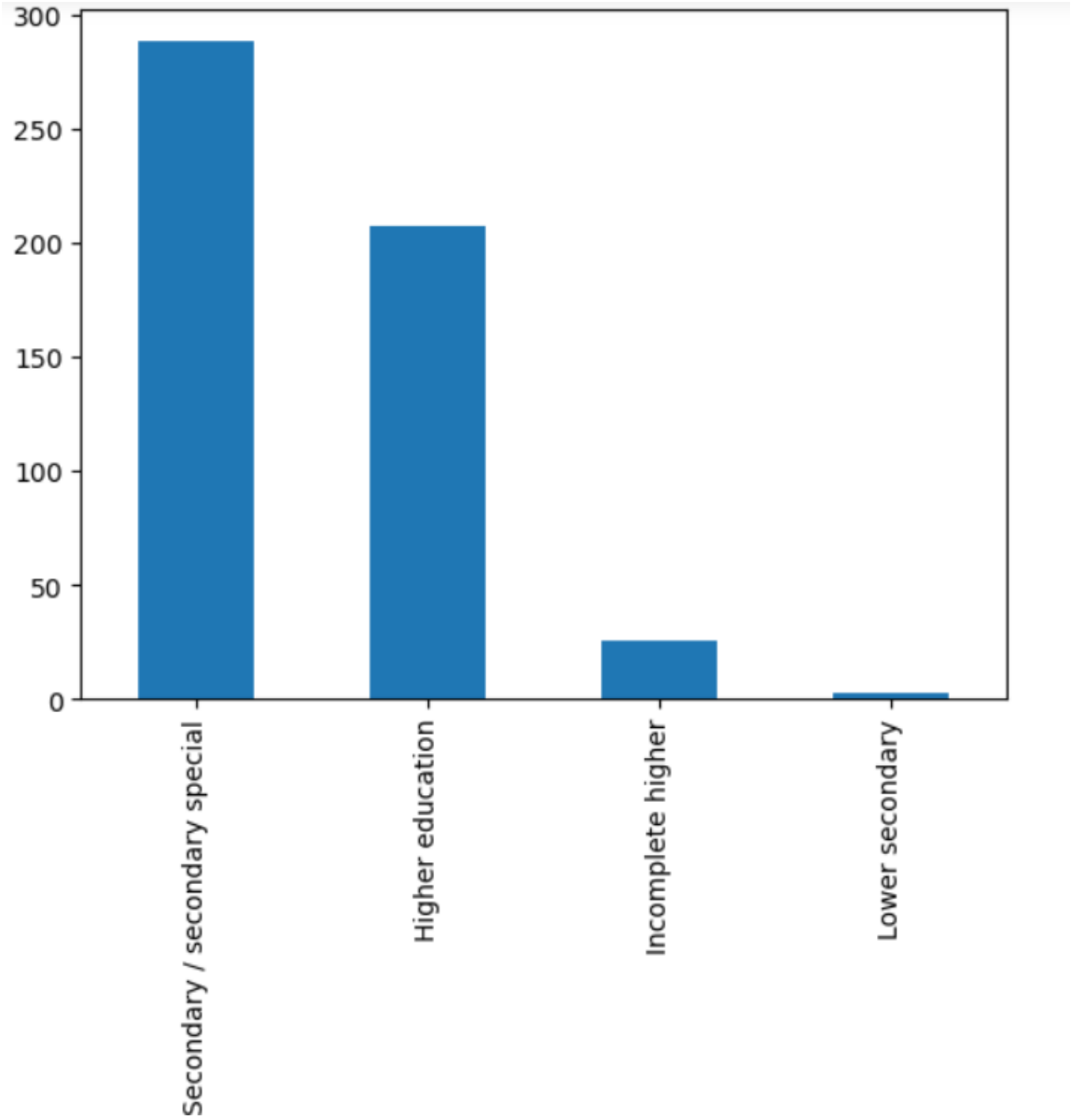
```
location_id    0
street name    0
city           0
state          0
zipcode        0
dtype: int64
```

Table Language:

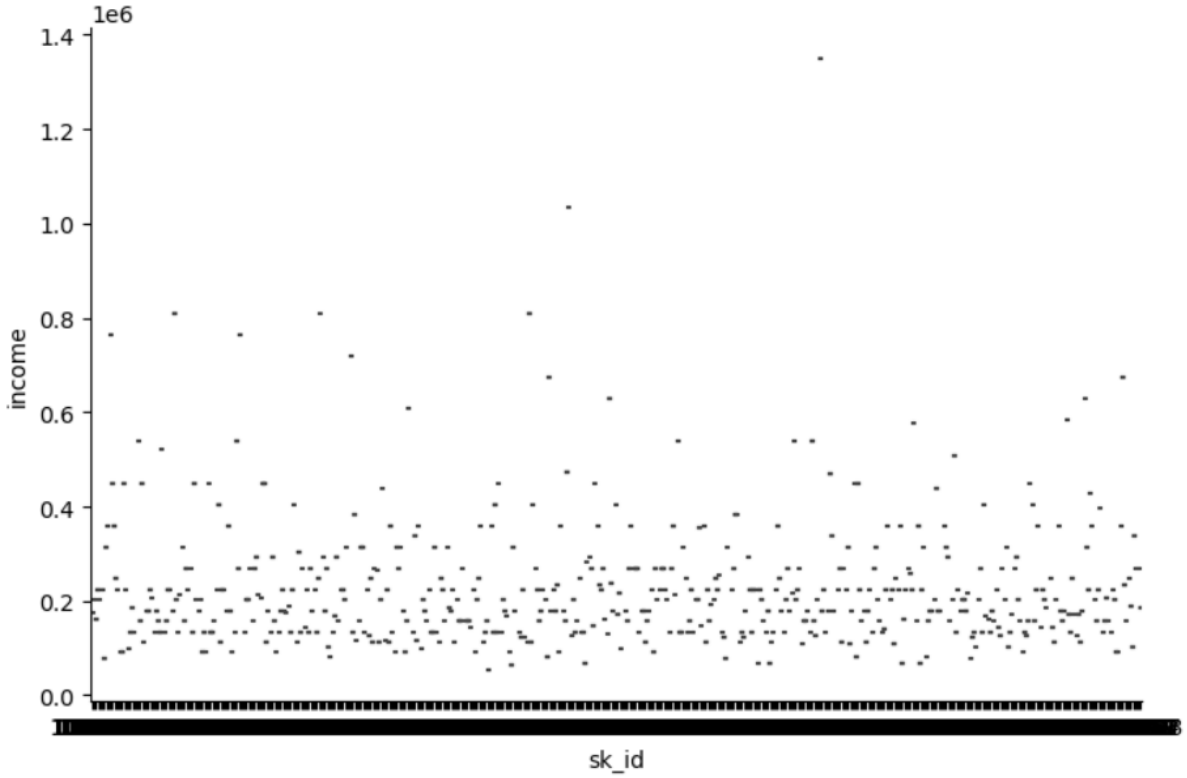
```
language.isnull().sum()
```

```
language_id     0
abbreviation    0
description     0
dtype: int64
```

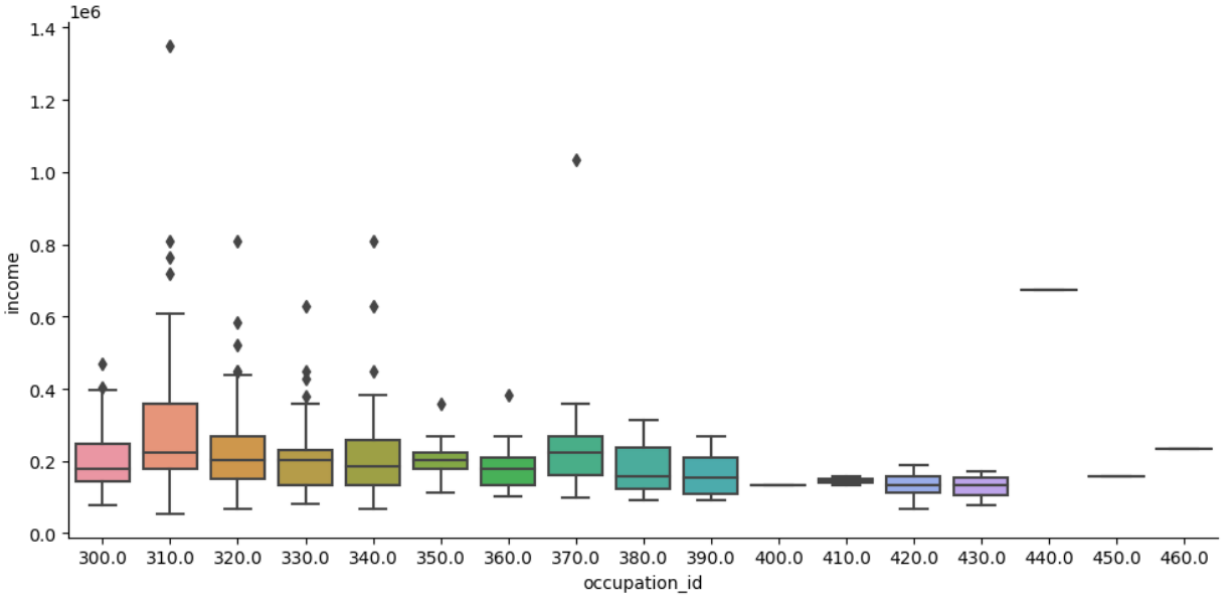
Visualizations:



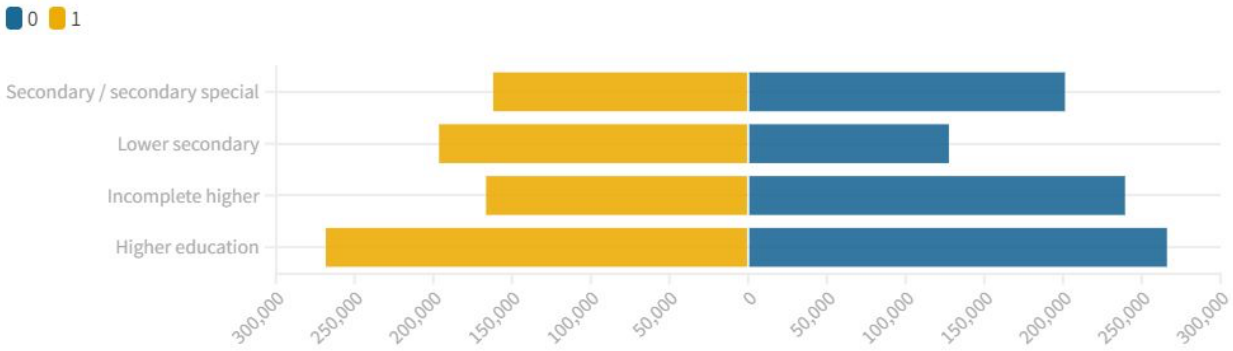
Maximum of our customers belong to Secondary/secondary special.



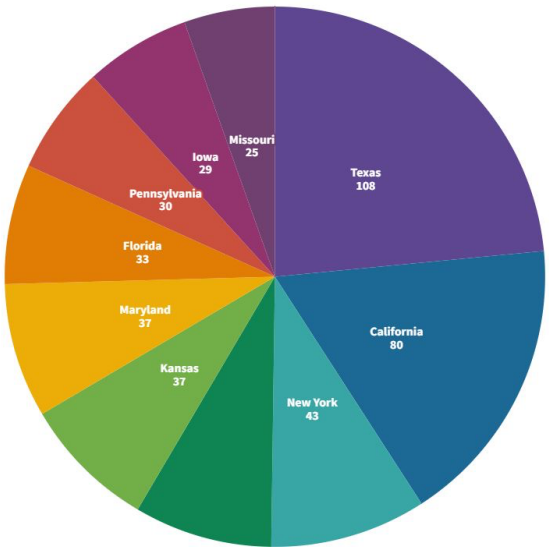
Our customer's salary mostly lie in the range of 1 lakhs to 6 lakhs annually.



The above graph displays the income distribution for each occupation_id.



The above graph shows the average income of the default or not default customers for each maximum education degree they have received. We see that Higher education customers that are not default receive higher income and lower secondary education customers that are not default receive less income than they are default.



The top 10 states that customers own property at. Texas and California are two major states where customers own property.

Sample snippets of Tables

Table creditloan:

```
1 •  select * from creditloan
2    limit 10;
```

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

ï»¿sk_id	target	income	education	occupation_id
100083	0	103500	Secondary / secondary special	300
100145	0	202500	Secondary / secondary special	300
100165	0	175500	Secondary / secondary special	
100179	0	202500	Higher education	310
100190	0	162000	Higher education	300
100193	0	225000	Secondary / secondary special	
100289	0	202500	Higher education	310
100295	1	225000	Secondary / secondary special	300
100341	0	76500	Secondary / secondary special	300
100343	0	315000	Secondary / secondary special	320

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. A relation will be in 3NF if it is in 2NF and no transition dependency exists.

Table Occupation:

```
4 •  select * from occupation
5    limit 10;
6
```

Result Grid | Filter Rows:

ï»¿occupation_id	role
300	Laborers
310	Managers
320	Drivers
330	Core Staff
340	Sales Staff
350	High Skill Tech Staff
360	Medicine staff
370	Accountants
380	Private service staff
390	Cooking staff

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. A relation will be in 3NF if it is in 2NF and no transition dependency exists.

Table twitter_data:

```
7 •  select * from twitter_data
8    limit 10;
```

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

Twitter_Id	Date_Published	Lik210	source_id	Tweet	Name	language_id	Retweets	200titi210	Lat210t24hours
1.15e18	2019-07-16 19:10:09+00:00	25	100	Do you ever get a wrinkled and nearly d210troy...	CreditLoan	200	2	{'hashtags': [{'text': 'dollarbill', 'indic210': [48, 5...	2022-11-13 11:05:17+00:0
1.15e18	2019-07-11 15:29:54+00:00	10	100	What are your plans after #highschool? Wheth...	CreditLoan	200	0	{'hashtags': [{'text': 'highschool', 'indic210': [26...	2022-11-13 10:34:08+00:0
1.15e18	2019-07-10 15:45:09+00:00	5	100	Whether you're trying to save up for something...	CreditLoan	200	0	{'hashtags': [{'text': 'debt', 'indic210': [76, 81]}...	2022-11-11 11:03:14+00:0
1.15e18	2019-07-09 17:20:00+00:00	2	100	Did your old car finally bite the dust? Ready for ...	CreditLoan	200	0	{'hashtags': [{'text': 'autoloans', 'indic210': [84,...	2022-11-11 10:41:18+00:0
1.15e18	2019-07-02 18:54:47+00:00	3	100	With all of the extra fe210 tacked onto an alrea...	CreditLoan	200	0	{'hashtags': [], 'symbols': [], 'user_m200tions': [...	2022-11-11 10:20:33+00:0
1.14e18	2019-06-28 18:33:12+00:00	1	100	Do you have a bad #creditscore? Sometim210 it...	CreditLoan	200	1	{'hashtags': [{'text': 'creditscore', 'indic210': [18...	2022-11-09 14:43:09+00:0
1.14e18	2019-06-19 18:41:15+00:00	1	100	Being replaced by a machine is a real fear in the...	CreditLoan	200	0	{'hashtags': [], 'symbols': [], 'user_m200tions': [...	2022-11-08 14:54:26+00:0
1.14e18	2019-06-12 19:04:40+00:00	2	100	High School Grads: Do NOT go off to #college w...	CreditLoan	200	0	{'hashtags': [{'text': 'college', 'indic210': [36, 44...	2022-11-08 11:04:48+00:0
1.14e18	2019-06-12 19:04:01+00:00	1	100	#HighSchool #Grads: Do NOT go off to #colleg...	CreditLoan	200	0	{'hashtags': [{'text': 'HighSchool', 'indic210': [0, ...	2022-11-08 10:32:21+00:0
1.14e18	2019-06-10 16:10:04+00:00	1	100	If you have yet to start an emerg200cy fund, ...	CreditLoan	200	1	{'hashtags': [], 'symbols': [], 'user_m200tions': [...	2022-11-08 09:44:24+00:0

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. A relation will be in 3NF if it is in 2NF and no transition dependency exists.

Table source:

```
10 •  select * from source
11    limit 10;
12
```

Result Grid | Filter Rows:

ï»¿source_id	title
100	Hootsuite Inc.
110	Twitter Web Client
120	Twitter for Websites
130	Quorum Government Relations

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. A relation will be in 3NF if it is in 2NF and no transition dependency exists.

Table language:

```
13 •  select * from language
14    limit 10;
15
```

Result Grid | Filter Rows:

ï»¿language_id	abbreviation	description
200	en	english
210	es	spanish
220	zh	chinese
230	fr	french

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. A relation will be in 3NF if it is in 2NF and no transition dependency exists.

Table Property:

```
16 •  select * from property
17    limit 10;
18
19
```

Result Grid | Filter Rows:

i»¿sk_id	location_id
100083	1320
100145	1160
100165	900
100179	1180
100190	1010
100193	1270
100289	940
100295	1330
100341	1260
100343	1330

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. A relation will be in 3NF if it is in 2NF and no transition dependency exists.

Table location:

```
19 •  select * from location
20    limit 10;
21
```

Result Grid | Filter Rows: | Export: | Wrap

location_id	street name	city	state	zipcode
800	Granby	Yakima	Washington	98907
810	Donald	El Paso	Texas	88530
820	Morrow	Memphis	Tennessee	38104
830	Donald	Lincoln	Nebraska	68583
840	Sullivan	Minneapolis	Minnesota	55487
850	Swallow	Lubbock	Texas	79491
860	Rutledge	Corpus Christi	Texas	78470
870	Hoard	Hartford	Connecticut	6183
880	Marcy	Miami	Florida	33124
890	Nevada	New York City	New York	10120

The table above is in 1NF as there are no multivalued attribute. It is also in 2NF as it is in 1NF and all non-key attributes are fully functional dependent on the primary key. In the above case, the streetname, city, state column is dependent on the zipcode column and the zipcode column is dependent on location_id.

The above scenario is called transitive dependency of the streetname, city, state columns on the location_id i.e. the primary key.
Hence we split the location as for the 3NF:

locates (location_id, zipcode)

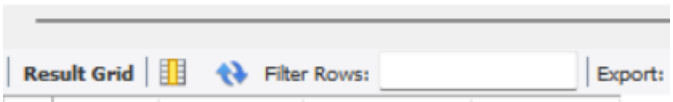
```
19 •  select * from locates
20    limit 10;
```

Result Grid | Filter Rows:

location_id	zipcode
800	98907
810	88530
820	38104
830	68583
840	55487
850	79491
860	78470
870	6183
880	33124
890	10120

zipcode (streetname, city, state, zipcode)

```
22 •  select * from zipcode
23    limit 10;
24
```



zipcode	streetname	city	state
98907	Granby	Yakima	Washington
88530	Donald	El Paso	Texas
38104	Morrow	Memphis	Tennessee
68583	Donald	Lincoln	Nebraska
55487	Sullivan	Minneapolis	Minnesota
79491	Swallow	Lubbock	Texas
78470	Rutledge	Corpus Christi	Texas
6183	Hoard	Hartford	Connecticut
33124	Marcy	Miami	Florida
10120	Nevada	New York City	New York

USE CASES

1st Use Case

Description: What tweets have been liked the most?
Actor:

Precondition: Likes should be more than 1

Steps:

Actor action: Admin checks for which are the most popular category

System Responses: System will check which is the most selling point of card.

Post Condition:

Alternate Path: Finding the number of retweets

Error: No error possible

Sql –

```
#Most Liked Tweet
select twitter_id, max(likes) as Most_Liked, Tweet
from student.creditfile;
```

Relational algebra-

π twitter_id, MAX (likes) → most_liked, tweet
γ MAX (likes) creditfile

2nd Use Case

Description: Languages in which tweets are posted
Actor:

Precondition: There should be one or more language offered in application

Steps:

Actor action: Admin checks for which are the most popular language across the application

System Responses: System can give recommendation based on the language.

Post Condition: Most popular language will display.

Error: No error possible

Sql

```
select language,count(twitter_id)
from student.creditfile
group by Language;
```

Relational algebra

γ language, COUNT (twitter_id) creditfile

3rd Use Case

Description: Most popular category in Credit card line

Precondition: Different credit cards must be provided to choose from

Actor action: Admin can update or delete the credit card according to the popularity and feedback

System Responses: System can give recommendation based on the language.

Post Condition: Most popular language will display.

Error: No error possible

Sql-

```
select tweet, max(retweets) as Retweeted
from student.creditfile;
```

Relational Algebra

π tweet, MAX (retweets) → retweeted
γ MAX (retweets) creditfile

4th Use Case

Description: Different Application process across the platform

Precondition: There should be number of application types

Actor action: Admin can choose how the application must work

System Responses: System will show different process which is available at this point of time.

Error: May give error if important parts are left

Sql-

```
select twitter_id as Twitter_ID, count(twitter_id) as Number_of_Tweets
from student.creditfile
group by twitter_id
limit 1;
```

Relational Algebra-

π twitter_id → twitter_id, COUNT (twitter_id) → number_of_tweets
γ twitter_id, COUNT (twitter_id) creditfile

5th Use Case

Description: Popularity of a particular credit card in different cities

Precondition: credit card should be offered in different cities

Actor action: Admin can launch new cities and can stop service too

System Responses: System will show the time left in different cities

Error: May not be able to show output if proper city is not selected

Sql-

```
select location, count(Twitter_id) as Total_Tweets
from student.creditfile
group by location;
```

Relational Algebra-

π location, COUNT (twitter_id) → total_tweets
γ location, COUNT (twitter_id) creditfile

6TH Use Case

Use Case: Count of customers for each occupation

Description: Giving the top occupations which are popular

Actor: User

Precondition: When a customer wants to buy something from shop, firstly he will be registered

Steps:

Actor action: User should have applied for credit

System Responses: System will provide the details of occupation if it is trustworthy or not.

Post Condition: Most popular occupation will receive credit

Alternate Path: The customer request is not correct and system throws an error

Error: User information is incorrect

```
select count(c.ï»¿sk_id) as count_of_customers, o.role
from creditloan c, occupation o
where c.occupation_id=o.ï»¿occupation_id
group by o.role
order by count(c.ï»¿sk_id) desc
limit 10;
```

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

count_of_customers	role
102	Laborers
87	Managers
47	Drivers
43	Core Staff
43	Sales Staff
23	High Skill Tech Staff
21	Accountants
16	Medicine staff
8	Private service staff
7	Security staff

7th Use case

Use Case: People whose average income is more than rest of cities

Description: Average income of some cities are more compared to others and there is possibility to get credit faster compared to other cities

Actor: User

Precondition: the customer should be from one of the cities in which average income is high

Steps:

Actor action: User request for credit

System Responses: if the client fulfills all the prerequisite the loan will be sanctioned

Post Condition: Customer need to maintain high income

Alternate Path: there will be no alternate path

```
8 •  select avg(c.income) as avg_income, l.state
9    from creditloan c, location l
10   where c.ï»¿sk_id=l.sk_id
11   group by l.state
12   order by avg(income) desc
13   limit 10;
14
15
16
17
```

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

avg_income	state
301090.9091	Colorado
265263.1579	District of Columbia
259071.4286	Oklahoma
249750.0000	California
246093.7500	Virginia
240827.5862	Iowa
239931.8182	Florida
238860.0000	Missouri
238500.0000	Utah
234395.8333	Texas

8th Use Case

Use Case: which language the tweets were posted from the states

Description: Language popular in each state where number of tweets were more.

Actor: User

Precondition: Must have logged in the account

Steps:

System Responses: Tweet should be atleast one time

Post Condition: Highest number of tweets must be posted

Alternate Path: There will be no alternate path

```
15 •  select la.description,l.state, count(*) as count_of_tweets
16    from  twitter_data t, creditloan c, location l,language la
17    where t.sk_id=c.ï»¿sk_id
18    and c.ï»¿sk_id=l.sk_id
19    and la.ï»¿language_id=t.language_id
20    group by la.description, l.state;
```

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

description	state	count_of_tweets
english	Illinois	4
french	Illinois	1
spanish	Wisconsin	1
spanish	Texas	3
chinese	North Carolina	1
english	North Carolina	1
spanish	North Carolina	1
english	California	17
english	Texas	33
english	Indiana	1
spanish	Indiana	1
chinese	California	1
english	Nebraska	5
english	Washington	7
french	Texas	1

9th Use Case

Use Case: What is the source in which tweets were posted in each state.

Description: Source popular in each state where number of tweets were more.

Actor: Title

Precondition: Must have logged in the account

Steps:
System Responses: Tweet should be posted more than one time

Post Condition: Highest number of tweets must be posted

Alternate Path: There will be no alternate path

```
22 •  select s.title,l.state, count(*) as count_of_tweets
23    from  twitter_data t, creditloan c, location l,source s
24    where t.sk_id=c.ï»¿sk_id
25    and c.ï»¿sk_id=l.sk_id
26    and s.ï»¿source_id=t.source_id
27    group by s.title, l.state;
28
```

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

title	state	count_of_tweets
Hootsuite Inc.	Illinois	1
Twitter Web Client	Illinois	4
Twitter Web Client	Wisconsin	3
Twitter Web Client	Texas	27
Twitter Web Client	North Carolina	3
Hootsuite Inc.	California	7
Twitter Web Client	California	12
Hootsuite Inc.	Texas	13
Hootsuite Inc.	Indiana	1
Twitter Web Client	Indiana	1
Hootsuite Inc.	Nebraska	2
Twitter Web Client	Nebraska	3
Hootsuite Inc.	Washington	2
Twitter Web Client	Washington	6
Hootsuite Inc.	Virginia	3

10th Use Case

Use Case: The average income for occupation in each case of default or not default
Description: User views the orders made by him/her

Actor: User

Precondition: Customer has either been defaulted or not defaulted

Steps:
System Responses: Displays all the income according to the role

```
29 •  select o.role, c.target, avg(c.income) as average_income
30    from occupation o, creditloan c
31    where c.occupation_id=o.ï»¿occupation_id
32    group by o.role, c.target;
33
```

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

role	target	average_income
Laborers	0	200571.4286
Managers	0	305470.5882
Laborers	1	196875.0000
Drivers	0	247600.0000
Core Staff	0	214123.1707
Sales Staff	0	227013.1579
Sales Staff	1	181800.0000
High Skill Tech Staff	0	205977.2727
Medicine staff	0	191892.8571
Accountants	0	258000.0000
Private service staff	1	90000.0000
Medicine staff	1	135000.0000
Private service staff	0	192857.1429
Cooking staff	0	166500.0000
HR staff	0	135000.0000
High Skill Tech Staff	1	265500.0000
Cleaning staff	1	157500.0000

Views for Use Cases

Query-1

```
1  Create View query_1 as
2  select count(c.ï»¿sk_id) as count_of_customers, o.role
3  from creditloan c, occupation o
4  where c.occupation_id=o.ï»¿occupation_id
5  group by o.role
6  order by count(c.ï»¿sk_id) desc
7  limit 10;
8  select * from query_1;
9
10
11
```

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

count_of_customers	role
102	Laborers
87	Managers
47	Drivers
43	Core Staff
43	Sales Staff
23	High Skill Tech Staff
21	Accountants
16	Medicine staff
8	Private service staff
7	Security staff

query 17

Query-2

```
10 Create view query_2 as
11 select z.state, avg(c.income) as avg_income
12 from creditloan c, property p, locates l, zipcode z
13 where c.ï»¿sk_id=p.ï»¿sk_id
14 and p.location_id=l.location_id
15 and l.zipcode=z.zipcode
16 group by z.state
17 order by avg(income) desc
18 limit 10;
19 select * from query_2;
20
21
```

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

state	avg_income
Colorado	301090.9091
Nevada	286714.2857
District of Columbia	262687.5000
New Jersey	248142.8571
Virginia	246093.7500
Pennsylvania	244057.5000
North Carolina	241568.1818
Iowa	240827.5862
Utah	238500.0000
Maryland	237600.0000

Query-3

```
Limit to 1000 rows
21 •  Create View query_3 as
22    select la.description, z.state, count(*) as count_of_tweets
23    from  twitter_data t, creditloan c, property p, locates l, zipcode z, language la
24    where t.sk_id=c.ï»¿sk_id
25    and c.ï»¿sk_id=p.ï»¿sk_id
26    and p.location_id=l.location_id
27    and l.zipcode=z.zipcode
28    and la.ï»¿language_id=t.language_id
29    group by la.description, z.state;
30 •  select * from query_3;
```

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

description	state	count_of_tweets
english	Illinois	8
english	California	15
english	West Virginia	4
english	Pennsylvania	8
english	Nevada	6
english	Texas	25
english	North Carolina	3
english	District of Columbia	9
english	Iowa	14
english	Nebraska	5
english	Virginia	13
english	Missouri	15
english	Michigan	7
english	Washington	13

query_3 9

Query-4

Assignment-4*

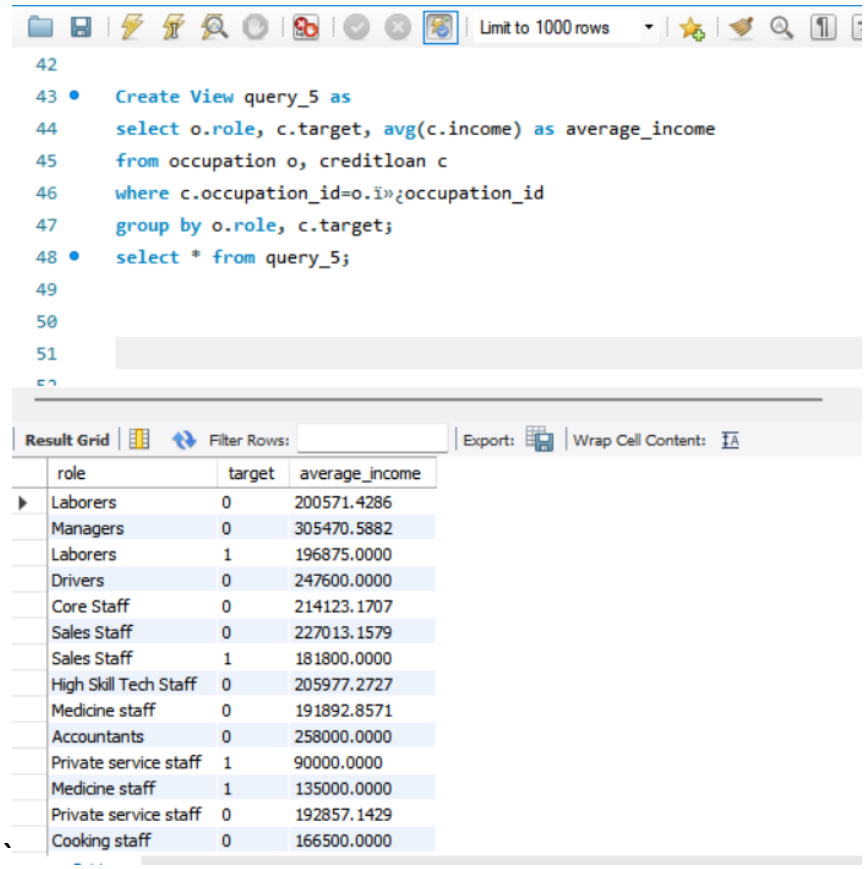
```
Limit to 1000 rows
32 •  Create View query_4 as
33    select s.title,z.state, count(*) as count_of_tweets
34    from  twitter_data t, creditloan c, property p, locates l, zipcode z, source s
35    where t.sk_id=c.ï»¿sk_id
36    and c.ï»¿sk_id=p.ï»¿sk_id
37    and p.location_id=l.location_id
38    and l.zipcode=z.zipcode
39    and s.ï»¿source_id=t.source_id
40    group by s.title, z.state;
41 •  select * from query_4;
```

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

title	state	count_of_tweets
Hootsuite Inc.	Illinois	4
Hootsuite Inc.	California	5
Hootsuite Inc.	West Virginia	2
Hootsuite Inc.	Pennsylvania	4
Hootsuite Inc.	Nevada	2
Hootsuite Inc.	Texas	9
Hootsuite Inc.	North Carolina	2
Hootsuite Inc.	District of Columbia	3
Hootsuite Inc.	Iowa	4
Hootsuite Inc.	Nebraska	2
Twitter Web ...	Illinois	4
Hootsuite Inc.	Virginia	3
Twitter Web ...	Virginia	11
Hootsuite Inc.	Missouri	4

query_4 10

Query-5



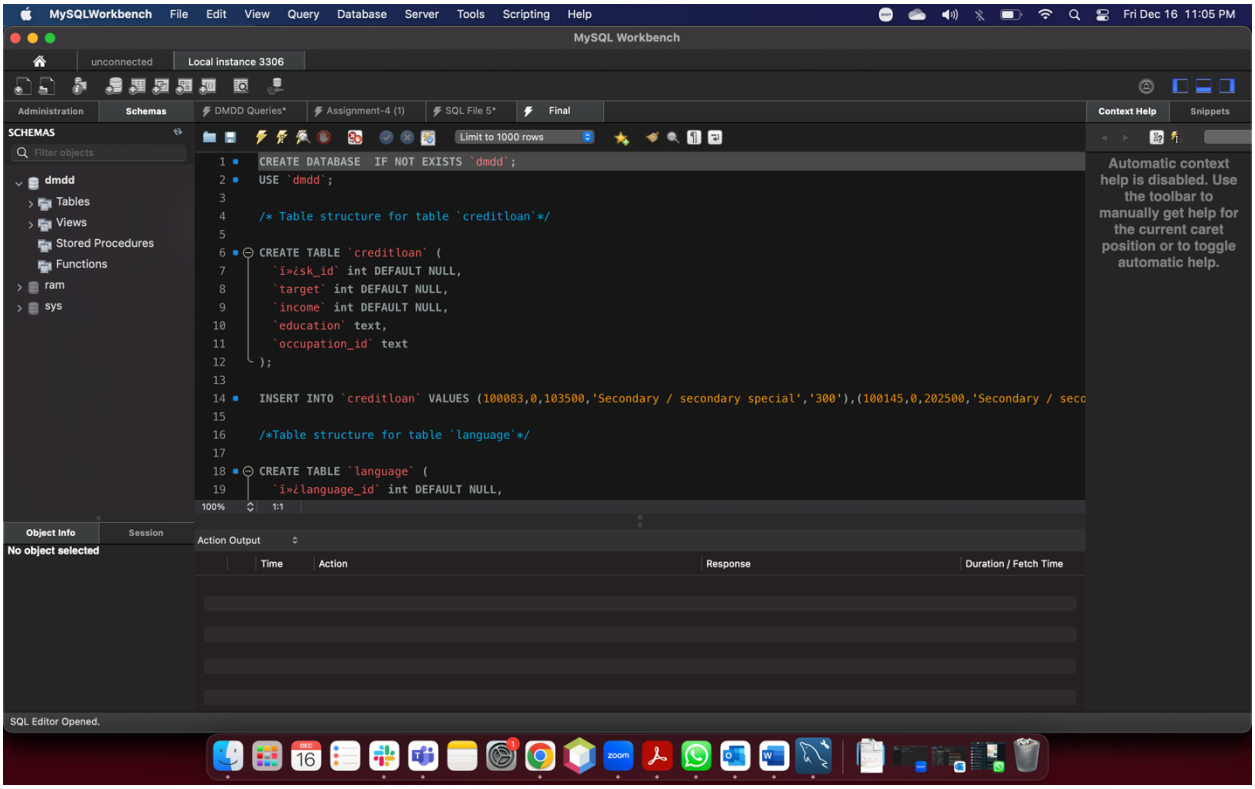
```
42
43 •  Create View query_5 as
44    select o.role, c.target, avg(c.income) as average_income
45    from occupation o, creditloan c
46    where c.occupation_id=o.ï»¿occupation_id
47    group by o.role, c.target;
48 •  select * from query_5;
49
50
51
```

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

role	target	average_income
Laborers	0	200571.4286
Managers	0	305470.5882
Laborers	1	196875.0000
Drivers	0	247600.0000
Core Staff	0	214123.1707
Sales Staff	0	227013.1579
Sales Staff	1	181800.0000
High Skill Tech Staff	0	205977.2727
Medicine staff	0	191892.8571
Accountants	0	258000.0000
Private service staff	1	90000.0000
Medicine staff	1	135000.0000
Private service staff	0	192857.1429
Cooking staff	0	166500.0000

Creation of Data Table

This is data schema which was created in MySQl Workbench. Multiple tables were created. Below is the example of the Data Schema of Credit Loan Data



Steps to reach the final database:

1. A credit loan dataset was chosen from kaggle. The data was cleaned and the columns that had null values was based on foreign key concept.
2. The twitter data was scrapped and linked to credit loan dataset with the random sk_ids.
3. The language and source columns was categorized to make the dataset more readable and understandable as languages and sources were just abbreviated.
4. Normalization was performed so to reduce the redundancies and more tables were created.