</talentlabs>

Credit Risk DA Project

Database Connection

Download the DBeaver SQL client to connect to the MySQL database:

- https://dbeaver.io/

Follow the documentation to set up a connection to the database:

- https://dbeaver.com/docs/wiki/Create-Connection/

The database is hosted on AWS, here are the connection details:

- Endpoint: home-credit-default-risk.c7rizeij2t53.ap-southeast-1.rds.amazonaws.com
- Port: 3306
- Database: credit
- Login User: student
- Login Password: student

Overview

Consider you are asked to review a list of loan applications. The given "credit" database contains data on the loan applicant and their historical loan behavior. There are many columns in the database, you **don't need to use all the columns**, we will provide a list of useful column descriptions for you.

Cautions

Missing Values:

There are columns with missing values. You need to handle them during your analysis. There are multiple ways we can handle missing values: 4 Ways to Replace NULL with a Different Value in MySQL

Discretization:

Discretization means we want to convert numbers into bins, for example, age to age groups or income to income groups. There are mainly 2 reasons for this:

- It is easier to see patterns with a group of values. For example, it is better to say people older than 20 are richer than people younger than 20, instead of saying people aged 20 are richer than people aged 21.
- We want to avoid biased statistics. If we apply group by aggregation directly on a number column like age, the average statistics can be biased. For example, if there is only 1 person aged 59, then the average income of people aged 59 only represents that 1 person in the dataset.

We can do it with the CASE Function in MySQL:
MySQL CASE Function
During the analysis, you can consider converting some factors into groups.

Task 1 Run SQL via DBeaver

Follow the documentation to open the "SQL Editor":

- https://dbeaver.com/docs/wiki/SQL-Editor/

Run SQL to examine the number of rows in each table:

Table	**Count**
application	307,511
bureau	1,716,428

Loan Applications

The "application" table stores the loan applications. This includes:

- The demographic of the loan applicants
- The loan size or purposes
- The applicant's credit score
- Is the loan applicant has a payment difficulties with the loan.

SK_ID_CURR	ID of the loan in our sample
TARGET	Target variable, this is the **future information**. Will this loan applicant has payment difficulties? (1: client with payment difficulties: he/she had late payment more than X days, 0: no payment difficulties)
CODE_GENDER	Gender of the client
FLAG_OWN_CAR	Flag if the client owns a car
FLAG_OWN_REALTY	Flag if the client owns a house or flat

CNT_CHILDREN	Number of children the client has
AMT_INCOME_TOTAL	Income of the client
AMT_CREDIT	Credit amount of the loan
AMT_ANNUITY	Loan annuity
AMT_GOODS_PRICE	For consumer loans it is the price of the goods for which the loan is given
NAME_TYPE_SUITE	Who was accompanying client when he was applying for the loan
NAME_INCOME_TYPE	Clients income type (businessman, working, maternity leave,…)
NAME_EDUCATION_TYPE	Level of highest education the client achieved
NAME_FAMILY_STATUS	Family status of the client
NAME_HOUSING_TYPE	What is the housing situation of the client (renting, living with parents, ...)
DAYS_BIRTH	Client's age in days at the time of application
DAYS_EMPLOYED	How many days before the application the person started current employment
OCCUPATION_TYPE	What kind of occupation does the client have
EXT_SOURCE_1	Normalized credit score from an external data source
EXT_SOURCE_2	Normalized credit score from an external data source
EXT_SOURCE_3	Normalized credit score from an external data source

Task 2 What is a Credit Score

In the “application” table above there are 3 credit score columns. Research online to see what is a credit score and why we need it. (Note that the scores in the database are normalized, which means they are scaled to the 0 to 1 range)

A **credit score** is a numerical representation of a person's creditworthiness, indicating how likely they are to repay borrowed funds. It is derived from various factors in a person's credit history and helps lenders assess risk when considering a loan application.

Task 3 Understand Credit Amount and Annuity

What are Credit Amount and Annuity? Fill in your answer below:

Credit Amount	**Credit Amount** refers to the total sum of money that a borrower is authorized to borrow from a lender. It can encompass various types of loans, such as personal loans, mortgages, or credit lines. The credit amount can be influenced by factors like the borrower's credit score, income, and repayment history. Lenders evaluate these factors to determine how much credit they are willing to extend to the borrower.
Annuity	**An annuity** is a financial product that provides a series of payments made at equal intervals. Annuities are commonly used for retirement planning, where individuals can invest a lump sum of money and receive periodic payments over time.

Task 4 Deduce the Loan Duration

Given the information from Task 4, we should be able to deduce the Loan Duration for each application. Loan duration describes how many periods (months) the applicant will need to pay back their loans.

Paste the SQL and part of the results below:

```
SELECT
    SK_ID_CURR,
    TARGET,
    AMT_CREDIT,
    AMT_ANNUITY,
    CASE
        WHEN AMT_ANNUITY > 0 AND AMT_CREDIT > 0 THEN ROUND(AMT_CREDIT /
AMT_ANNUITY, 2)
        ELSE NULL
    END AS Loan_Duration_Months
FROM
    application
WHERE
    AMT_CREDIT IS NOT NULL
    AND AMT_ANNUITY IS NOT NULL;
```

Task 5 Are there any factors in the application table affecting the Credit Scores?

In the "application" table try to explore if there are any columns affecting the credit score. For example, is gender a factor?

Do the analysis of at least 3 factors for 3 different credit scores, it is expected to see different results for different credit scores, for example, a factor might affect EXT_SOURCE_1 but not EXT_SOURCE_3.

Please explain your findings with SQL statements and results:

1. Impact of Gender on Credit Scores

```
SELECT
    CASE
        WHEN CODE_GENDER = 'F' THEN 'Female'
        WHEN CODE_GENDER = 'M' THEN 'Male'
        ELSE 'Other/Unknown'
    END AS Gender,
    COUNT(*) AS Total_Applicants,
    SUM(CASE WHEN EXT_SOURCE_1 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_1,
    SUM(CASE WHEN EXT_SOURCE_2 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_2,
    SUM(CASE WHEN EXT_SOURCE_3 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_3,
    ROUND(AVG(COALESCE(EXT_SOURCE_1, 0)), 4) AS Avg_Ext_Source_1,
    ROUND(AVG(COALESCE(EXT_SOURCE_2, 0)), 4) AS Avg_Ext_Source_2,
    ROUND(AVG(COALESCE(EXT_SOURCE_3, 0)), 4) AS Avg_Ext_Source_3
FROM
    application
WHERE
    CODE_GENDER IS NOT NULL
GROUP BY
    CASE
        WHEN CODE_GENDER = 'F' THEN 'Female'
        WHEN CODE_GENDER = 'M' THEN 'Male'
        ELSE 'Other/Unknown'
    END;
```

Output:

Gender	Total_Applicants	Total_Null_Ext_Source_1	Total_Null_Ext_Source_2	Total_Null_Ext_Source_3	Avg_Ext_Source_1	Avg_Ext_Source_2	Avg_Ext_Source_3
Male	105,059	62,568	158	22,352	0.1646	0.5097	0.3972
Female	202,448	110,808	502	38,613	0.2472	0.5151	0.416
Other/Unknown	4	2	0	0	0.2645	0.5808	0.3025

2. Impact of Income Type on Credit Scores

```
SELECT
    NAME_FAMILY_STATUS AS Family_Status,
    COUNT(*) AS Total_Applicants,
    SUM(CASE WHEN EXT_SOURCE_1 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_1,
    SUM(CASE WHEN EXT_SOURCE_2 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_2,
    SUM(CASE WHEN EXT_SOURCE_3 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_3,
```

```
    ROUND(AVG(COALESCE(EXT_SOURCE_1, 0)), 4) AS Avg_Ext_Source_1,
    ROUND(AVG(COALESCE(EXT_SOURCE_2, 0)), 4) AS Avg_Ext_Source_2,
    ROUND(AVG(COALESCE(EXT_SOURCE_3, 0)), 4) AS Avg_Ext_Source_3
FROM
    application
WHERE
    NAME_FAMILY_STATUS IS NOT NULL
GROUP BY
    Family_Status
ORDER BY
    Total_Applicants DESC;
```

Output:

Income_Type	Total_Applicants	Total_Null_Ext_Source_1	Total_Null_Ext_Source_2	Total_Null_Ext_Source_3	Avg_Ext_Source_1	Avg_Ext_Source_2	Avg_Ext_Source_3
Working	158,774	84,455	315	31,780	0.2185	0.5009	0.3987
Commercial associate	71,617	35,700	141	14,871	0.251	0.5411	0.3962
Pensioner	55,362	42,512	156	11,053	0.1609	0.505	0.4473
State servant	21,703	10,680	48	3,229	0.2656	0.5334	0.4378
Unemployed	22	15	0	17	0.1326	0.4593	0.1054
Student	18	9	0	4	0.2103	0.5	0.4175
Businessman	10	6	0	6	0.2088	0.6664	0.2252
Maternity leave	5	1	0	5	0.5329	0.5121	0

3. Impact of Family Status on Credit Scores

```
SELECT
    NAME_FAMILY_STATUS AS Family_Status,
    COUNT(*) AS Total_Applicants,
    SUM(CASE WHEN EXT_SOURCE_1 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_1,
    SUM(CASE WHEN EXT_SOURCE_2 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_2,
    SUM(CASE WHEN EXT_SOURCE_3 IS NULL THEN 1 ELSE 0 END) AS Total_Null_Ext_Source_3,
    ROUND(AVG(COALESCE(EXT_SOURCE_1, 0)), 4) AS Avg_Ext_Source_1,
    ROUND(AVG(COALESCE(EXT_SOURCE_2, 0)), 4) AS Avg_Ext_Source_2,
    ROUND(AVG(COALESCE(EXT_SOURCE_3, 0)), 4) AS Avg_Ext_Source_3
FROM
    application
WHERE
    NAME_FAMILY_STATUS IS NOT NULL
GROUP BY
    Family_Status
ORDER BY
    Total_Applicants DESC;
```

Output:

Family_Status	Total_Applicants	Total_Null_Ext_Source_1	Total_Null_Ext_Source_2	Total_Null_Ext_Source_3	Avg_Ext_Source_1	Avg_Ext_Source_2	Avg_Ext_Source_3
Married	196,432	110,244	388	36,084	0.2245	0.5193	0.4194
Single / not married	45,444	24,186	105	11,009	0.2056	0.4959	0.3706
Civil marriage	29,775	16,435	83	7,072	0.2106	0.4964	0.3827
Separated	19,770	10,797	35	3,621	0.2412	0.5231	0.419
Widow	16,088	11,715	49	3,178	0.178	0.5086	0.4379
Unknown	2	1	0	1	0.3143	0.6729	0.3353

Task 6 Are there any factors in the application table affecting the Credit Amount?

Who is going to lend more money than others? In this task, we want to see are there any factors affecting the credit amount. **Do the analysis of at least 3 factors**

</talentlabs>

Please explain your findings with SQL statements and results:

1. **Analysis by Income Level**

```sql
SELECT
    CASE
        WHEN AMT_INCOME_TOTAL IS NULL OR AMT_INCOME_TOTAL < 0 THEN 'Invalid Income'
        WHEN AMT_INCOME_TOTAL BETWEEN 0 AND 9999 THEN '0 - 9,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 10000 AND 19999 THEN '10,000 - 19,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 20000 AND 29999 THEN '20,000 - 29,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 30000 AND 39999 THEN '30,000 - 39,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 40000 AND 49999 THEN '40,000 - 49,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 50000 AND 59999 THEN '50,000 - 59,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 60000 AND 69999 THEN '60,000 - 69,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 70000 AND 79999 THEN '70,000 - 79,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 80000 AND 89999 THEN '80,000 - 89,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 90000 AND 99999 THEN '90,000 - 99,999'
        ELSE '100,000 and above'
    END AS Income_Level,
    COUNT(*) AS Total_Applicants,
    ROUND(AVG(COALESCE(AMT_CREDIT, 0)), 4) AS Average_Amount_Credit,
    ROUND(MAX(COALESCE(AMT_CREDIT, 0)), 4) AS Max_Amount_Credit,
    ROUND(MIN(COALESCE(AMT_CREDIT, 0)), 4) AS Min_Amount_Credit,
    ROUND(STDDEV(COALESCE(AMT_CREDIT, 0)), 4) AS Std_Dev_Amount_Credit
FROM
    application
WHERE
    AMT_INCOME_TOTAL IS NOT NULL AND AMT_INCOME_TOTAL >= 0
GROUP BY
    Income_Level
ORDER BY
     MIN(AMT_INCOME_TOTAL);
```

Output

Income_Level	Total_Applicants	Average_Amount_Credit	Max_Amount_Credit	Min_Amount_Credit	Std_Dev_Amount_Credit
20,000 - 29,999	141	283,338.9255	2,173,500	45,000	257,457.0652
30,000 - 39,999	822	284,708.1734	1,223,010	45,000	206,460.5012
40,000 - 49,999	3,554	297,421.9035	2,215,224	45,000	224,063.1072
50,000 - 59,999	4,035	318,388.374	2,146,500	45,000	239,241.4922
60,000 - 69,999	12,881	346,182.747	2,205,000	45,000	246,349.4028
70,000 - 79,999	5,930	382,589.0605	2,013,840	45,000	260,698.615
80,000 - 89,999	8,076	395,821.8076	1,800,000	45,000	269,094.1699
90,000 - 99,999	28,259	429,052.2866	2,517,300	45,000	284,745.5614
100,000 and above	243,813	654,362.9346	4,050,000	45,000	412,903.9687

2. **Analysis by Car Ownership**

```sql
SELECT
    COALESCE(FLAG_OWN_CAR, 'Unknown') AS Car_Own,
    COUNT(*) AS Total_Applicants,
    ROUND(AVG(COALESCE(AMT_CREDIT, 0)), 4) AS Avg_Amount_Credit,
    ROUND(STDDEV(COALESCE(AMT_CREDIT, 0)), 4) AS Std_Dev_Amount_Credit,
    ROUND(MIN(COALESCE(AMT_CREDIT, 0)), 4) AS Min_Amount_Credit,
    ROUND(MAX(COALESCE(AMT_CREDIT, 0)), 4) AS Max_Amount_Credit
FROM
```

</talentlabs>

```
    application
GROUP BY
    Car_Own
ORDER BY
    Car_Own;
```

Output

Car_Own	Total_Applicants	Avg_Amount_Credit	Std_Dev_Amount_Credit	Min_Amount_Credit	Max_Amount_Credit
N	202,924	565,442.541	380,921.5439	45,000	4,050,000
Y	104,587	664,186.008	433,991.4468	45,000	4,050,000

3. **Analysis by Family Status**

```
SELECT
    NAME_FAMILY_STATUS AS Family_Status,
    COUNT(*) AS Total_Applicants,
    ROUND(AVG(COALESCE(AMT_CREDIT, 0)), 4) AS Avg_Amt_Credit,
    ROUND(MAX(COALESCE(AMT_CREDIT, 0)), 4) AS Max_Amt_Credit,
    ROUND(MIN(COALESCE(AMT_CREDIT, 0)), 4) AS Min_Amt_Credit,
    ROUND(STDDEV(COALESCE(AMT_CREDIT, 0)), 4) AS Std_Dev_Amt_Credit
FROM
    application
GROUP BY
    NAME_FAMILY_STATUS  -- Removed extra comma here
ORDER BY
    Avg_Amt_Credit DESC;
```

Output

Family_Status	Total_Applicants	Avg_Amt_Credit	Max_Amt_Credit	Min_Amt_Credit	Std_Dev_Amt_Credit
Married	196,432	642,999.7908	4,050,000	45,000	419,430.6324
Unknown	2	630,000	675,000	585,000	45,000
Separated	19,770	552,113.8204	3,150,000	45,000	373,869.2685
Civil marriage	29,775	541,573.4596	4,050,000	45,000	362,070.8973
Single / not married	45,444	505,350.1788	4,031,032.5	45,000	354,284.64
Widow	16,088	490,695.9086	2,576,898	45,000	332,583.5886

Task 7 Are there any factors in the application table affecting the Payment Difficulties?

In the database, the TARGET column describes will there be a payment difficulty for a loan. We want to see if there are any factors in the application table that can be used to predict this future information. **Do the analysis of at least 3 factors**

Please explain your findings with SQL statements and results:

1. **Analysis by Employment Length**

```
SELECT
```

```
    CASE
        WHEN DAYS_EMPLOYED IS NULL THEN 'Unknown'
        WHEN DAYS_EMPLOYED < -1000 THEN 'Unemployed'
        WHEN DAYS_EMPLOYED BETWEEN -1000 AND -730 THEN 'Short-Term Employment (1-2
years)'
        WHEN DAYS_EMPLOYED BETWEEN -729 AND -365 THEN 'Short-Term Employment (Less
than 1 year)'
        WHEN DAYS_EMPLOYED BETWEEN -364 AND -183 THEN 'Currently Employed (6 months
- 1 year)'
        WHEN DAYS_EMPLOYED BETWEEN -182 AND 0 THEN 'Currently Employed (Less than 6
months)'
        WHEN DAYS_EMPLOYED BETWEEN 1 AND 183 THEN 'New Employment (Less than 6
months)'
        ELSE 'Long-Term Employment (More than 1 year)'
    END AS Employment_Status,
    COUNT(*) AS Total_Applicants,
    SUM(TARGET) AS Total_Payment_Difficulties,
    ROUND(AVG(TARGET), 2) AS Avg_Payment_Difficulty,
    ROUND(AVG(TARGET) * 100, 2) AS Percentage_Payment_Difficulties
FROM
    application
WHERE
    TARGET IS NOT NULL
GROUP BY
    Employment_Status
ORDER BY
    Avg_Payment_Difficulty DESC;
```

Output

Employment_Status	Total_Applicants	Total_Payment_Difficulties	Avg_Payment_Difficulty	Percentage_Payment_Difficulties
Short-Term Employment (Less than 1 year)	31,841	3,631	0.11	11.4
Currently Employed (6 months - 1 year)	18,152	2,031	0.11	11.19
Short-Term Employment (1-2 years)	22,453	2,470	0.11	11
Currently Employed (Less than 6 months)	9,752	1,034	0.11	10.6
Unemployed	169,939	12,669	0.07	7.46
Long-Term Employment (More than 1 year)	55,374	2,990	0.05	5.4

2. Analysis by Family Status

```
SELECT
    NAME_FAMILY_STATUS AS Family_Status,
    COUNT(*) AS Total_Applicants,
    ROUND(SUM(TARGET), 2) AS Total_Payment_Difficulties,
    ROUND(AVG(TARGET), 2) AS Avg_Payment_Difficulty,
    ROUND(AVG(TARGET) * 100, 2) AS Percentage_Payment_Difficulties
FROM
    application
WHERE
    TARGET IS NOT NULL
GROUP BY
    Family_Status
ORDER BY
    Percentage_Payment_Difficulties DESC;
```

Output

Family_Status	Total_Applicants	Total_Payment_Difficulties	Avg_Payment_Difficulty	Percentage_Payment_Difficulties
Civil marriage	29,775	2,961	0.1	9.94
Single / not married	45,444	4,457	0.1	9.81
Separated	19,770	1,620	0.08	8.19
Married	196,432	14,850	0.08	7.56
Widow	16,088	937	0.06	5.82
Unknown	2	0	0	0

3. Analysis by Income Level

```sql
SELECT
    CASE
        WHEN AMT_INCOME_TOTAL IS NULL OR AMT_INCOME_TOTAL < 0 THEN 'Invalid Income'
        WHEN AMT_INCOME_TOTAL BETWEEN 0 AND 9999 THEN '0 - 9,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 10000 AND 19999 THEN '10,000 - 19,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 20000 AND 29999 THEN '20,000 - 29,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 30000 AND 39999 THEN '30,000 - 39,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 40000 AND 49999 THEN '40,000 - 49,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 50000 AND 59999 THEN '50,000 - 59,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 60000 AND 69999 THEN '60,000 - 69,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 70000 AND 79999 THEN '70,000 - 79,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 80000 AND 89999 THEN '80,000 - 89,999'
        WHEN AMT_INCOME_TOTAL BETWEEN 90000 AND 99999 THEN '90,000 - 99,999'
        ELSE '100,000 and above'
    END AS Income_Level,
    COUNT(*) AS Total_Applicants,
    ROUND(AVG(AMT_INCOME_TOTAL), 2) AS Avg_Income,
    MIN(AMT_INCOME_TOTAL) AS Min_Income,
    MAX(AMT_INCOME_TOTAL) AS Max_Income,
    SUM(TARGET) AS Total_Payment_Difficulties,
    ROUND(SUM(TARGET) * 100.0 / COUNT(*), 2) AS Percentage_Payment_Difficulties,
    ROUND(AVG(TARGET), 2) AS Avg_Payment_Difficulty
FROM
    application
GROUP BY
    Income_Level
ORDER BY
    MIN(AMT_INCOME_TOTAL);
```

Output

Income_Level	Total_Applicants	Avg_Income	Min_Income	Max_Income	Total_Payment_Difficulties	Percentage_Payment_Difficulties	Avg_Payment_Difficulty
20,000 - 29,999	141	27,827.55	25,650	29,700	8	5.67	0.06
30,000 - 39,999	822	34,933.6	30,150	39,600	72	8.76	0.09
40,000 - 49,999	3,554	45,489.05	40,050	49,950	263	7.4	0.07
50,000 - 59,999	4,035	55,416.44	50,085	59,850	295	7.31	0.07
60,000 - 69,999	12,881	66,980.33	60,075	69,943.5	1,050	8.15	0.08
,999	5,930	74,765.27	70,024.5	79,875	481	8.11	0.08
80,000 - 89,999	8,076	82,125.15	80,032.5	89,995.5	721	8.93	0.09
90,000 - 99,999	28,259	91,506.3	90,000	99,900	2,335	8.26	0.08
100,000 and above	243,813	192,500.21	100,071	117,000,000	19,600	8.04	0.08

Previous/Other Loan Applications

In the previous section, we explored if the demographic data related to payment difficulties, this section we want to see if **historical loan behavior** affecting the payment difficulties.

The “bureau” table stores the other loans of the applicants from the other lenders.

“bureau” table:

SK_ID_CURR	ID of loan in our sample - one loan in our sample can have 0,1,2 or more related previous credits in credit bureau
SK_BUREAU_ID	Recoded ID of previous Credit Bureau credit related to our loan (unique coding for each loan application), The IDs of the “other loans”
CREDIT_DAY_OVERDUE	Number of days past due on CB credit at the time of application for related loan in our sample
AMT_CREDIT_MAX_OVERDUE	Maximal amount overdue on the Credit Bureau credit so far (at application date of loan in our sample)
CNT_CREDIT_PROLONG	How many times was the Credit Bureau credit prolonged
AMT_CREDIT_SUM	Current credit amount for the Credit Bureau credit
AMT_CREDIT_SUM_DEBT	Current debt on Credit Bureau credit
AMT_CREDIT_SUM_LIMIT	Current credit limit of credit card reported in Credit Bureau
AMT_CREDIT_SUM_OVERDUE	Current amount overdue on Credit Bureau credit
CREDIT_TYPE	Type of Credit Bureau credit (Car, cash,...)
DAYS_CREDIT_UPDATE	How many days before loan application did last information about the Credit Bureau credit come
AMT_ANNUITY	Annuity of the Credit Bureau credit

</talentlabs>

Task 7 Is the number of other loans affecting the payment difficulties?

We want to see if loan applicants have other historical loans affecting their payment abilities.
Hints:

- You will need to count the number of loans for each SK_ID_CURR in the "bureau" table.
- Transform the counts into count groups (Discretization).
- Compute the relation between average other loan count to the TARGET

Paste the SQL and part of the results below:

```sql
SELECT
    CASE
        WHEN Total_Loan = 0 THEN '0 Loans'
        WHEN Total_Loan BETWEEN 1 AND 2 THEN '1-2 Loans'
        WHEN Total_Loan BETWEEN 3 AND 4 THEN '3-4 Loans'
        WHEN Total_Loan BETWEEN 5 AND 6 THEN '5-6 Loans'
        ELSE '7+ Loans'
    END AS Loan_Group,
    AVG(a.TARGET) * 100 AS Average_Payment_Difficulty_Percent,
    COUNT(a.SK_ID_CURR) AS Number_of_Applicants
FROM (
    SELECT
        SK_ID_CURR,
        COUNT(*) AS Total_Loan
    FROM
        bureau
    GROUP BY
        SK_ID_CURR
) AS Total_Loans
JOIN
    application a ON Total_Loans.SK_ID_CURR = a.SK_ID_CURR
GROUP BY
    Loan_Group
ORDER BY
    Loan_Group;
```

Output:

Loan_Group	Average_Payment_Difficulty_Percent	Number_of_Applicants
1-2 Loans	8.2028	71,707
3-4 Loans	7.3831	61,898
5-6 Loans	7.3361	45,937
7+ Loans	7.7976	83,949

</talentlabs>

Task 8 FreeStyle

Now, conduct your own research and analysis to see what factors from the “application” and the “bureau” tables are affecting

- The Credit Scores
- The Payment Difficulty