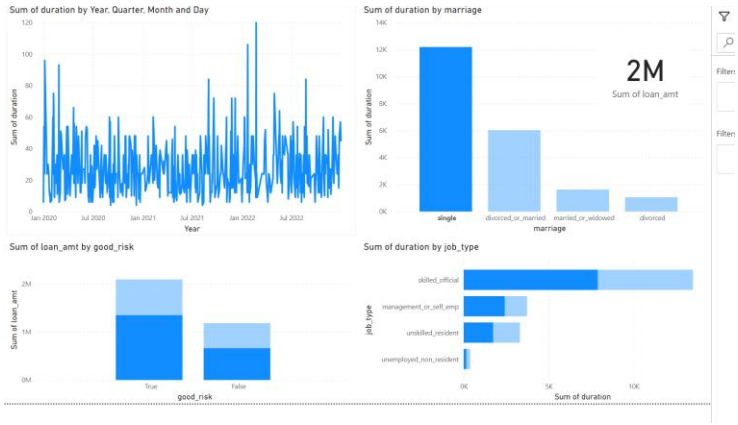
PartA

Extra data summarization on PowerBI (you can find others in the code)

The sum of loan amount for all applicant with management or self-employed job type (attribute)

The sum of loan amount for all applicant with single marriage status (attribute)



The sum of loan amount with good risk (attribute)

Part A report:

Process data in fact table and joined table. The data processing code performs several key preprocessing steps on a dataset containing numerical features. Below is a summary of the preprocessing steps along with any data transformation and quality issues encountered:

Handling missing values:

No missing values found.

Handling categorical attributes through e.g., one-hot encoding:

One hot encoding for categorial attributes like: saving account, checking account, purpose

Normalization using Min-Max Scaling:

The initial dataset contains 7 numerical columns that are unsuitable for direct use in machine learning models due to lack of normalization.

Min-Max Scaling is applied using MinMaxScaler() to scale the numerical features to a range between 0 and 1.

This scaling ensures that features with larger magnitudes (e.g., salary) do not dominate the learning process compared to features with smaller magnitudes (e.g., age or loan duration).

Data Transformation:

Before normalization, descriptive statistics are provided for the original dataset, including minimum, maximum, mean, and standard deviation values for each numerical feature.

After normalization, a similar table is generated, displaying the minimum, maximum, mean, and standard deviation values for each feature, now scaled between 0 and 1.

Final Dataset:

After preprocessing, the dataset is updated with the normalized numerical columns, resulting in a cleaner dataset ready for further analysis or modeling.

PartB

Skipped (Verified with TA)

PartC :

We used a One-Class SVM algorithm from Scikit-Learn, setting its "nu" parameter to 0.01.By setting nu to 0.01, we aimed to catch the most extreme outliers, about 1% of our data, focusing on the biggest differences. This setting helps because our data doesn't vary much, making it tough to find real outliers. So, we changed the usual setting from 0.5 to 0.01 to target only the most unusual data points. Here's how we did it:

```
from sklearn.svm import OneClassSVM
svm = OneClassSVM(nu=0.01, gamma='auto').fit(data)
outliers = data[svm.predict(data) == -1]
```

After running this, we found 17 outliers out of 1000 data points. An interesting find was the higher number of non-foreign workers among these outliers compared to their overall presence in the dataset. While only about 3.7%(37/1000) of all the data points were non-foreign workers, they made up about 11.75%(2/17) of the outliers. This difference suggests there might be specific reasons why the financial data of non-foreign workers stands out. Also, finding guarantors with these outliers was uncommon, indicating special financial situations.

Overall, the variables with the largest proportion difference were non-foreign workers as well as borrowers with guarantors. This finding suggests potential biases or unique circumstances affecting non-foreign workers' financial behavior, possibly influencing their representation in outlier data. Moreover, the presence of borrowers with guarantors among outliers underlines another dimension of atypical financial activity, as this characteristic is exceptionally rare in the dataset. Using the One-Class SVM helped us not just identify outliers but also learn more about certain traits within our data. This understanding is crucial for investigating further or adjusting our financial models to reflect these unique cases. It shows why finding outliers is important in data science—it helps us understand our data better and spot potential issues that could affect our models.