

## Use SQL to process and join data:

### Have a quick view of the data:

1. Select 100 records from each table to understand the data/variables
  - a. e.g. select \* from application limit 100
2. Use describe table to understand each column's format
3. Identify the primary key in each table used to join tables
4. Check how many total records in each table, and how many distinct records (use the primary key to identify distinct records)
5. How many accounts in each CREDIT\_TYPE in **Bureau** table?
6. Use sk\_id\_curr=215354 as an example to collect the following columns:
  - a. All columns from **APPLICATION**
  - b. SK\_BUREAU\_ID,CREDIT\_ACTIVE,CREDIT\_CURRENCY,DAYS\_CREDIT,AMT\_CREDIT\_SUM,AMT\_CREDIT\_SUM\_DEBT,CREDIT\_TYPE from **Bureau**
  - c. You have to use **JOIN**

## Part One: Explore and process Application data:

*For every question, think out why it is asked from business perspective, and you can export the data to Excel to create the chart:*

7. Check the distribution of AMT\_CREDIT, which is the credit limit applied by each customer

8. Check the distribution of loan repaid or not (use the Target variable)



9. Check the distribution of loan type (NAME\_CONTRACT\_TYPE)

10. Create the following new columns and think about whether it makes sense:

- $NEW\_CREDIT\_TO\_GOODS\_RATIO = AMT\_CREDIT / AMT\_GOODS\_PRICE$
- $NEW\_CAR\_TO\_BIRTH\_RATIO = OWN\_CAR\_AGE / DAYS\_BIRTH$
- $NEW\_CREDIT\_TO\_INCOME\_RATIO = AMT\_CREDIT / AMT\_INCOME\_TOTAL$

## Part Two: Explore and process Bureau data:

11. check out how many distinct SK\_ID\_CURR group by different credit status (CREDIT\_ACTIVE) and different CREDIT\_TYPE

12. Do aggregation calculations for columns by SK\_ID\_CURR by different CREDIT\_ACTIVE each as below:

- DAYS\_CREDIT : [ min , max , avg ] /\* let's pick one variable days\_credit first\*/
- CREDIT\_DAY\_OVERDUE : [ max , avg ]

13. You may realize that each SK\_ID\_CURR has multiple records based on different values of CREDIT\_ACTIVE, so we need to reshape the data to ensure one record per SK\_ID\_CURR => This is commonly called as 'flatten the dataset'

14. create additional 2 columns – whether the customer has bad debt before (**bd\_flag** = 1 or 0), how many previous bad debts the customer has (**bd\_num**)

### Explore and process Bureau balance data:

15. Check distinct status

16. Create new columns:

- Status\_0\_mean: average number of status=0 for each SK\_ID\_BUREAU
- Status\_1\_mean: average number of status=1 for each SK\_ID\_BUREAU
- Status\_2\_mean: average number of status=2 for each SK\_ID\_BUREAU
- Status\_3\_mean: average number of status=3 for each SK\_ID\_BUREAU
- Status\_4\_mean: average number of status=4 for each SK\_ID\_BUREAU
- Status\_5\_mean: average number of status=5 for each SK\_ID\_BUREAU
- Status\_c\_mean: average number of status=c for each SK\_ID\_BUREAU
- Status\_x\_mean: average number of status=x for each SK\_ID\_BUREAU

17. Find the corresponding SK\_ID\_CURR in Application table for each SK\_ID\_BUREAU

- You need Bureau table as an intermedium table to complete the job
- Be careful! One SK\_ID\_CURR may have multiple SK\_ID\_BUREAU, but in the end, you may want to generate the result which has one record per SK\_ID\_CURR and the mean of each status for each SK\_ID\_CURR:

SK_ID_CURR	status_c_mean	status_x_mean	status_0_mean	status_1_mean	status_2_mean	status_3_mean	status_4_mean	status_5_mean
100002	0.2091	0.1364	0.4091	0.2455	0.0000	0.0000	0.0000	0.0000
100010	0.7222	0.0000	0.2778	0.0000	0.0000	0.0000	0.0000	0.0000
100019	0.0000	0.0588	0.9412	0.0000	0.0000	0.0000	0.0000	0.0000
100032	0.0000	0.1053	0.8947	0.0000	0.0000	0.0000	0.0000	0.0000
100033	0.0000	0.8571	0.1429	0.0000	0.0000	0.0000	0.0000	0.0000

*Hint: you may want to join first, then do the case when and group by*

### Explore and process Previous application data:

18. Check whether one SK\_ID\_CURR may have multiple SK\_ID\_PREV => this will affect the join results

19. Create a new column APP\_CREDIT\_PERC = AMT\_APPLICATION / AMT\_CREDIT for approved decisions: value ask / value received percentage

20. Check distinct NAME\_CONTRACT\_STATUS, and then add 2 new columns for each SK\_ID\_CURR:

- Num\_of\_app (the number of total approved previous applications; *Think about how to deal with 'Unused offer' status*)
- Num\_of\_ref

21. For approved and used accounts, calculate the average of APP\_CREDIT\_PERC (avg\_APP\_CREDIT\_PERC) and aggregate to one record per SK\_ID\_CURR

22. Let's join Application table with Bureau table, Bureau Balance, and Previous Application together with all new added columns! (from Bureau table, we only need the new added columns)

