Trabajo de Consultas

Presentado por Carlos Dubón [@AmadeusCelta]

Preparación del ámbiente

In [2]:
```
# Librerías a Utilizar
import pandas as pd
from datetime import timedelta
```

Preparación de los datos

In [3]:
```
# Cargar las tablas
credit_record = pd.read_csv("credit_record.csv")  # Contiene Id, StatusDate, Status, DepositBalance
loan_applications = pd.read_csv("loan_applications.csv")  # Contiene Id, Application_Date
```

In [7]:
```
# Renombrar la primera columna (actualente sin nombre) como 'Correlative'
credit_record.rename(columns={credit_record.columns[0]: 'Correlative'}, inplace=True)
loan_applications.rename(columns={loan_applications.columns[0]: 'Correlative'}, inplace=True)
```

In [10]:
```
# Convertir las fechas a datetime
credit_record['Status_date'] = pd.to_datetime(credit_record['Status_date'])
loan_applications['application_date'] = pd.to_datetime(loan_applications['application_date'])
```

In [11]:
```
credit_record
```

Out[11]:

	Correlative	ID	Status_date	STATUS	saldo_depositos
0	0	5008804	2023-01-26	C	516.19
1	1	5008804	2022-12-26	C	13190.50
2	2	5008804	2022-11-26	C	204.67
3	3	5008804	2022-10-26	C	1214.29
4	4	5008804	2022-09-26	C	335.20
...	...	...	...	...	...
777710	777710	5150487	2021-06-14	C	29653.00
777711	777711	5150487	2021-05-15	C	292.47
777712	777712	5150487	2021-04-14	C	717.48
777713	777713	5150487	2021-03-15	C	5631.45
777714	777714	5150487	2021-02-12	C	2661.42

777715 rows × 5 columns

In [12]:
```
loan_applications
```

Out[12]:

	Correlative	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INC
0	0	5008804	M	Y	Y	0	
1	1	5008805	M	Y	Y	0	
2	2	5008806	M	Y	Y	0	
3	3	5008808	F	N	Y	0	
4	4	5008809	F	N	Y	0	
...	...	...	...	...	...	...	
438458	438552	6840104	M	N	Y	0	
438459	438553	6840222	F	N	N	0	
438460	438554	6841878	F	N	N	0	
438461	438555	6842765	F	N	Y	0	
438462	438556	6842885	F	N	Y	0	

438463 rows × 20 columns

Procesamiento de la información

In [18]:
```
# Paso 1: Obtener estados al cierre de cada mes
credit_record['YearMonth'] = credit_record['Status_date'].dt.to_period('M')  # Crear agrupación Año-M
credit_record['Row'] = credit_record.groupby(['ID', 'YearMonth'])['Status_date'].rank(ascending=False
record_status = credit_record[credit_record['Row'] == 1][['ID', 'Status_date', 'STATUS']]  # Filtrar

record_status
```

Out[18]:

	ID	Status_date	STATUS
0	5008804	2023-01-26	C
1	5008804	2022-12-26	C
2	5008804	2022-11-26	C
3	5008804	2022-10-26	C
4	5008804	2022-09-26	C
...	...	...	...
777710	5150487	2021-06-14	C
777711	5150487	2021-05-15	C
777712	5150487	2021-04-14	C
777713	5150487	2021-03-15	C
777714	5150487	2021-02-12	C

768904 rows × 3 columns

In [19]:
```
# Paso 2: Obtener el estado según la fecha de aplicación (-3 meses)
record_status['EOMONTH_StatusDate'] = record_status['Status_date'].dt.to_period('M').dt.to_timestamp(
loan_applications['EOMONTH_ApplicationDate'] = (loan_applications['application_date'] - pd.offsets.Mo
customer_status = record_status.merge(
    loan_applications[['ID', 'EOMONTH_ApplicationDate']],
    left_on=['ID', 'EOMONTH_StatusDate'],
    right_on=['ID', 'EOMONTH_ApplicationDate'],
    how='inner'
)[['ID', 'Status_date', 'STATUS']]  # Relación entre STATUS y fecha de

customer_status
```

Out[19]:

	ID	Status_date	STATUS
0	5008805	2023-03-05	C
1	5008806	2021-07-04	X
2	5008812	2023-04-25	X
3	5008814	2022-12-29	0
4	5008821	2022-02-10	X
...	...	...	...
13181	5150475	2022-07-29	C
13182	5150477	2023-04-18	0
13183	5150478	2021-11-27	C
13184	5150481	2022-02-26	X
13185	5150487	2021-12-14	C

13186 rows × 3 columns

In [20]:
```
# Paso 3: Obtener registros para promedios (últimos 6 meses)
loan_applications['START_DATE'] = loan_applications['application_date'] - pd.DateOffset(months=6)  #
record_balance = credit_record.merge(
    loan_applications[['ID', 'application_date', 'START_DATE']],
```

```
    on='ID',
    how='inner'
)
record_balance_filtered = record_balance[
    (record_balance['Status_date'] >= record_balance['START_DATE']) &
    (record_balance['Status_date'] < record_balance['application_date'])
]  # Filtrar rango de 6 meses

record_balance_filtered
```

Out[20]:

	Correlative	ID	Status_date	STATUS	saldo_depositos	YearMonth	Row	application_date	STAR
21	21	5008805	2023-06-05	C	726.88	2023-06	1.0	2023-06-06	2022
22	22	5008805	2023-05-05	C	1180.00	2023-05	1.0	2023-06-06	2022
23	23	5008805	2023-04-05	C	891.26	2023-04	1.0	2023-06-06	2022
24	24	5008805	2023-03-05	C	442.38	2023-03	1.0	2023-06-06	2022
25	25	5008805	2023-02-03	C	2622.67	2023-02	1.0	2023-06-06	2022
...	...	...	...	...	...	...	...	...	
777702	777702	5150487	2022-02-13	C	1017.75	2022-02	1.0	2022-03-20	2021
777703	777703	5150487	2022-01-13	C	279.59	2022-01	1.0	2022-03-20	2021
777704	777704	5150487	2021-12-14	C	183.83	2021-12	1.0	2022-03-20	2021
777705	777705	5150487	2021-11-13	C	1669.43	2021-11	1.0	2022-03-20	2021
777706	777706	5150487	2021-10-14	C	100.46	2021-10	1.0	2022-03-20	2021

79613 rows × 9 columns

In [29]:
```
# Paso 4: Calcular el promedio de los saldos
customer_balance = record_balance_filtered.groupby('ID')['saldo_depositos'].mean().round(2).reset_ind
customer_balance.rename(columns={'saldo_depositos': 'AVGDepositBalance'}, inplace=True)
customer_balance
```

Out[29]:

	ID	AVGDepositBalance
0	5008805	3769.74
1	5008806	1929.04
2	5008811	1567.59
3	5008812	15259.63
4	5008814	872.08
...	...	...
16757	5150475	544.40
16758	5150477	2493.24
16759	5150478	2197.64
16760	5150481	778.93
16761	5150487	1201.69

16762 rows × 2 columns

In [30]:
```
# Paso 5: Integrar resultados finales
final_result = loan_applications.merge(customer_status, on='ID', how='left') \
    .merge(customer_balance, on='ID', how='left')
final_result['STATUS'] = final_result['STATUS'].fillna('No disponible')  # Sustituir valores nulos en
```

Resultados

In [31]:
```
# Resultado final
final_result
```

Out[31]:

	Correlative	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INC
0	0	5008804	M	Y	Y	0	
1	1	5008805	M	Y	Y	0	
2	2	5008806	M	Y	Y	0	
3	3	5008808	F	N	Y	0	
4	4	5008809	F	N	Y	0	
...	...	...	...	...	...	...	
438458	438552	6840104	M	N	Y	0	
438459	438553	6840222	F	N	N	0	
438460	438554	6841878	F	N	N	0	
438461	438555	6842765	F	N	Y	0	
438462	438556	6842885	F	N	Y	0	

438463 rows × 25 columns