**--- Nulos y duplicados**

Select count(user\_id), count(age) as age, count(sex) as sex, count(last\_month\_salary) as month\_salary, count(number\_dependents) as number\_dependents FROM `banco-riesgo-relativo.datos\_banco.user\_info`;

Select count(\*)

FROM `banco-riesgo-relativo.datos\_banco.user\_info`

where number\_dependents is null;

Select age, count(\*)

FROM `banco-riesgo-relativo.datos\_banco.user\_info`

group by age

Having count(user\_id) > 0

order by age asc;

**---- Identificar y manejar datos fuera del alcance del análisis**

SELECT CORR (a.age, b.default\_flag) as correlation

FROM `banco-riesgo-relativo.datos\_banco.user\_info` as a

INNER JOIN `banco-riesgo-relativo.datos\_banco.default` as b

ON a.user\_id = b.user\_id;

WITH loans\_correction AS (

  SELECT

    CASE

      WHEN LOWER (loan\_type) LIKE 'real estate' THEN 'Real State'

      WHEN LOWER (loan\_type) LIKE 'other' THEN 'Other'

      WHEN LOWER(loan\_type) LIKE 'others' THEN 'Other'

    END AS loan\_type\_c

  FROM `banco-riesgo-relativo.datos\_banco.loans\_outstanding`)

  SELECT loan\_type\_c

  FROM loans\_correction

  GROUP BY loan\_type\_c;

**---Nueva vista para avg de salario y dependientes por edad**

SELECT

  CASE

    WHEN age < 20 THEN 19

    WHEN age BETWEEN 20 and 29 THEN 20

    WHEN age BETWEEN 30 and 39 THEN 30

    WHEN age BETWEEN 40 and 49 THEN 40

    WHEN age BETWEEN 50 and 59 THEN 50

    WHEN age BETWEEN 60 and 69 THEN 60

    WHEN age BETWEEN 70 and 79 THEN 70

    WHEN age BETWEEN 80 and 89 THEN 80

    ELSE 90

  END AS age\_category,

  CAST(round(AVG(number\_dependents),0) AS INT64) AS avg\_dependants,

  round(AVG(last\_month\_salary),2) as avg\_salary,

FROM `banco-riesgo-relativo.datos\_banco.user\_info`

GROUP BY age\_category

**---Identificar y tratar los datos OUTLIERS**

Select MIN(last\_month\_salary), MAX(last\_month\_salary), AVG(last\_month\_salary)

FROM `banco-riesgo-relativo.datos\_banco.user\_info`;

Select last\_month\_salary, user\_id, age

FROM `banco-riesgo-relativo.datos\_banco.user\_info`

order by last\_month\_salary desc;

---\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

WITH TukeyPercentiles AS (

  SELECT

    PERCENTILE\_CONT(last\_month\_salary, 0.02) OVER () AS lower\_percentile,

    PERCENTILE\_CONT(last\_month\_salary, 0.98) OVER () AS upper\_percentile

  FROM

    `banco-riesgo-relativo.datos\_banco.user\_info`

  LIMIT 1  -- Limitamos a 1 para asegurarnos de obtener solo un conjunto de percentiles

)

SELECT user\_id, last\_month\_salary

FROM  `banco-riesgo-relativo.datos\_banco.user\_info` AS u

JOIN TukeyPercentiles AS p

ON 1=1  -- Hacemos un join con 1=1 ya que solo necesitamos un conjunto de percentiles

WHERE last\_month\_salary >= p.lower\_percentile AND last\_month\_salary <= p.upper\_percentile

ORDER BY last\_month\_salary ASC;

**--- Nueva variables**

SELECT u.\*,

  CASE

    WHEN lo.user\_id IS NOT NULL THEN 1 ELSE 0

  END AS with\_bank\_credit,

  ---SUM(CASE WHEN lo.user\_id IS NOT NULL THEN 1 ELSE 0 END) OVER () AS total\_creditos

FROM `banco-riesgo-relativo.datos\_banco.user\_info` as u

LEFT JOIN `banco-riesgo-relativo.datos\_banco.loans\_outstanding` as lo

ON u.user\_id = lo.user\_id

order by with\_bank\_credit asc

**-- Nueva vista para totales de créditos por cliente.**

WITH loans\_correction AS (

  SELECT

    CASE

      WHEN LOWER (loan\_type) LIKE 'real estate' THEN 'Real State'

      WHEN LOWER (loan\_type) LIKE 'other' THEN 'Other'

      WHEN LOWER(loan\_type) LIKE 'others' THEN 'Other'

    END AS loan\_type\_c,

    user\_id

  FROM `banco-riesgo-relativo.datos\_banco.loans\_outstanding`),

loans\_qty\_rs AS (

  SELECT user\_id, count(\*) AS qty\_loans\_rs

  FROM loans\_correction

  WHERE loan\_type\_c = 'Real State'

  GROUP BY user\_id

),

loans\_qty\_ot AS (

  SELECT user\_id, count(\*) AS qty\_loans\_ot

  FROM loans\_correction

  WHERE loan\_type\_c = 'Other'

  GROUP BY user\_id

)

SELECT COALESCE(a.user\_id, b.user\_id) AS user\_id, COALESCE(a.qty\_loans\_rs,0) AS qty\_loans\_rs,  COALESCE(b.qty\_loans\_ot,0) AS qty\_loans\_ot,

COALESCE(a.qty\_loans\_rs, 0) + COALESCE(b.qty\_loans\_ot, 0) as total\_loans

FROM loans\_qty\_rs as a

FULL OUTER JOIN loans\_qty\_ot as b

ON a.user\_id = b.user\_id

ORDER BY a.user\_id asc;

**-- Outliers en Colab**

# Seleccionar solo las columnas a analizar

interested\_columns = ['using\_lines\_not\_secured\_personal\_assets', 'debt\_ratio', 'total\_delayed\_payments']

# Crear un nuevo DataFrame SOLO con las columnas seleccionadas

df\_subset\_loan\_detail = df\_loan\_detail[interested\_columns]

# Configurar el tamaño del gráfico

plt.figure(figsize=(12, 4))

# Crear un diccionario para almacenar la cantidad de outliers y los valores más extremos por variable

outliers\_info\_dict = {}

# Iterar sobre las columnas seleccionadas y crear subgráficos para boxplot y histograma

#Ciclo for para dar n vueltas y repetir las acciones dentro del for

for i, column in enumerate(interested\_columns, 1):

  # Calcular los percentiles para identificar outliers

  Q1 = df\_subset\_loan\_detail[column].quantile(0.25)

  Q2 = df\_subset\_loan\_detail[column].quantile(0.50)

  Q3 = df\_subset\_loan\_detail[column].quantile(0.75)

  IQR = Q3 - Q1

  # Definir límites para identificar outliers

  lower\_limit = Q1 - 1.5 \* IQR

  upper\_limit = Q3 + 1.5 \* IQR

  # Filtrar datos para identificar outliers

  outliers = df\_subset\_loan\_detail[(df\_subset\_loan\_detail[column] < lower\_limit) | (df\_subset\_loan\_detail[column] > upper\_limit)]

  print(outliers)

  # Almacenar información sobre outliers y valores más extremos

  outliers\_info\_dict[column] = {

      'q1\_value': Q1,

      'q2\_value': Q2,

      'q3\_value': Q3,

      'iqr\_value': IQR,

      'lower\_limit': lower\_limit,

      'upper\_limit': upper\_limit,

      'outliers\_count': len(outliers),

      'min\_value': df\_subset\_loan\_detail[column].min(),

      'max\_value': df\_subset\_loan\_detail[column].max(),

      'first\_outlier\_value': outliers[column].min() if not outliers.empty else None,

      'mean\_value': df\_subset\_loan\_detail[column].mean(),

      'median\_value': df\_subset\_loan\_detail[column].median()

  }

  # # CREACION DE NUEVO df PARA QUEDARNOS CON LOS VALORES FILTRADOS.

  # df\_subset\_loan\_detail = df\_subset\_loan\_detail[(df\_subset\_loan\_detail[column] >= lower\_limit) & (df\_subset\_loan\_detail[column] <= upper\_limit)]

  for variable, valor in outliers\_info\_dict[column].items():

    print(f'{variable}: {valor}')

  #Plot

  plt.figure(figsize=(12, 4))  # Tamano de la figura, largo por ancho

  sbn.boxplot(x=df\_subset\_loan\_detail[column], data=df\_subset\_loan\_detail)

  plt.title(f'Boxplot variable {column}')

  plt.xlabel(column)

  plt.ylabel('Frecuencia')

  plt.show()

  #Histograma

  plt.figure(figsize=(12, 4))

  plt.hist(df\_subset\_loan\_detail[column], bins=30, color='skyblue', edgecolor='black')

  plt.title(f'Histograma variable {column}')

  plt.xlabel(column)

  plt.ylabel('Frecuencia')

  plt.show()

# # Supongamos que outliers\_info\_dict es tu diccionario

# for columna, info in outliers\_info\_dict.items():

#   print(f'Información de la columna {columna}:')

#   for variable, valor in info.items():

#       print(f'{variable}: {valor}')

#   print('\n')  # Agrega una línea en blanco entre las columnas para mayor claridad

df\_subset\_loan\_detail.describe()

Ejemplo resultado:

A screenshot of a graph

Description automatically generated

**-- Vista de usuarios con aplicacion de Outliers**

SELECT u.\* EXCEPT (last\_month\_salary\_c),

CASE

  WHEN dependents\_c < 3 THEN CAST(dependents\_c AS INT64)

  WHEN dependents\_c >=3 THEN 3

END AS dependents\_category,

CASE

  WHEN last\_month\_salary\_c >= 13220 THEN 13220

  ELSE last\_month\_salary\_c

END AS last\_month\_salary\_cleaned

FROM  `banco-riesgo-relativo.datos\_banco.user\_info\_c` as u

WHERE

  age <= 94

ORDER BY user\_id ASC;

**-- Vista de loan\_detail con aplicacion de Outliers**

SELECT user\_id,

using\_lines\_not\_secured\_personal\_assets,

CASE

  WHEN using\_lines\_not\_secured\_personal\_assets > 1.327033046 THEN 1.327033046 ELSE using\_lines\_not\_secured\_personal\_assets

END AS using\_lines\_not\_secured\_personal\_assets\_cleaned,

debt\_ratio,

CASE

  WHEN debt\_ratio > 1.9194819487499999 THEN 1.9194819487499999 ELSE debt\_ratio

END AS debt\_ratio\_cleaned,

number\_times\_delayed\_payment\_loan\_30\_59\_days, number\_times\_delayed\_payment\_loan\_60\_89\_days,

number\_times\_delayed\_payment\_loan\_30\_59\_days+number\_times\_delayed\_payment\_loan\_60\_89\_days AS number\_times\_short\_delays,

more\_90\_days\_overdue,

number\_times\_delayed\_payment\_loan\_30\_59\_days + number\_times\_delayed\_payment\_loan\_60\_89\_days + more\_90\_days\_overdue AS total\_delayed\_payments

FROM `banco-riesgo-relativo.datos\_banco.loan\_detail`

**-- Union de Tablas , con nuevas variables y aplicación de categorías y cuartiles**

ELECT u.\* ,

CASE

  WHEN dependents\_category = 0 THEN '1. Sin Hijos'

  WHEN dependents\_category = 1 THEN '2. Con 1 Hijo'

  WHEN dependents\_category = 2 THEN '3. Con 2 Hijos'

  WHEN dependents\_category = 3 THEN '4. Con 3 o mas Hijos'

END AS dependents\_categoryname,

CASE

  WHEN age BETWEEN 20 AND 34 THEN 1

  WHEN age BETWEEN 35 AND 49 THEN 2

  WHEN age BETWEEN 50 AND 64 THEN 3

  WHEN age BETWEEN 65 AND 79 THEN 4

  WHEN age >= 80  THEN 5

END AS age\_qtl,

CASE

  WHEN age BETWEEN 20 AND 34 THEN '1. Joven'

  WHEN age BETWEEN 35 AND 49 THEN '2. Adulto Joven'

  WHEN age BETWEEN 50 AND 64 THEN '3. Adulto Mediana Edad'

  WHEN age BETWEEN 65 AND 79 THEN '4. Adulto Experimentado'

  WHEN age >= 80 THEN '5. Adulto Mayor'

END AS age\_categoryname,

NTILE(5) OVER (ORDER BY last\_month\_salary\_cleaned) AS salary\_qtl,

CASE

  WHEN NTILE(5) OVER (ORDER BY last\_month\_salary\_cleaned) =1 THEN '1. Bajo'

  WHEN NTILE(5) OVER (ORDER BY last\_month\_salary\_cleaned) =2 THEN '2. Moderado'

  WHEN NTILE(5) OVER (ORDER BY last\_month\_salary\_cleaned) =3 THEN '3. Medio'

  WHEN NTILE(5) OVER (ORDER BY last\_month\_salary\_cleaned) =4 THEN '4. Alto'

  WHEN NTILE(5) OVER (ORDER BY last\_month\_salary\_cleaned) =5 THEN '5. Muy Alto'

END AS salary\_category,

d.\* EXCEPT(user\_id),

IFNULL(lo.qty\_loans\_rs,0) AS qty\_loans\_rs,

IFNULL(lo.qty\_loans\_ot,0) AS qty\_loans\_ot,

IFNULL(lo.total\_loans,0) AS total\_loans,

NTILE(5) OVER (ORDER BY total\_loans) AS total\_loans\_qtl,

CASE

  WHEN NTILE(5) OVER (ORDER BY total\_loans) =1 THEN '1. Bajo'

  WHEN NTILE(5) OVER (ORDER BY total\_loans) =2 THEN '2. Moderado'

  WHEN NTILE(5) OVER (ORDER BY total\_loans) =3 THEN '3. Medio'

  WHEN NTILE(5) OVER (ORDER BY total\_loans) =4 THEN '4. Alto'

  WHEN NTILE(5) OVER (ORDER BY total\_loans) =5 THEN '5. Muy Alto'

END AS total\_loans\_category,

CASE

  WHEN lo.user\_id IS NOT NULL THEN 1 ELSE 0

END AS with\_bank\_credit,

ld.using\_lines\_not\_secured\_personal\_assets\_cleaned,

CASE

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .25 THEN 1

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .50 THEN 2

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .75 THEN 3

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .99 THEN 4

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned > .99 THEN 5

END AS using\_lines\_qtl,

CASE

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .25 THEN '1. Bajo'

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .50 THEN '2. Moderado'

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .75 THEN '3. Medio'

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned <= .99 THEN '4. Alto'

  WHEN ld.using\_lines\_not\_secured\_personal\_assets\_cleaned > .99 THEN '5. Excesivo'

END AS using\_lines\_category,

ld.debt\_ratio\_cleaned,

CASE

  WHEN ld.debt\_ratio\_cleaned <= .25 THEN 1

  WHEN ld.debt\_ratio\_cleaned <= .50 THEN 2

  WHEN ld.debt\_ratio\_cleaned <= .75 THEN 3

  WHEN ld.debt\_ratio\_cleaned <= .99 THEN 4

  WHEN ld.debt\_ratio\_cleaned > .99 THEN 5

END AS debt\_ratio\_qtl,

CASE

  WHEN ld.debt\_ratio\_cleaned <= .25 THEN '1. Bajo'

  WHEN ld.debt\_ratio\_cleaned <= .50 THEN '2. Moderado'

  WHEN ld.debt\_ratio\_cleaned <= .75 THEN '3. Medio'

  WHEN ld.debt\_ratio\_cleaned <= .99 THEN '4. Alto'

  WHEN ld.debt\_ratio\_cleaned > .99 THEN '5. Excesivo'

END AS debt\_ratio\_category,

ld.more\_90\_days\_overdue,

IF (ld.more\_90\_days\_overdue >=1, 1, 0) AS is\_90\_days\_overdue,

CASE

  WHEN ld.total\_delayed\_payments  >= 250 THEN 250 ELSE ld.total\_delayed\_payments

END AS total\_delayed\_payments\_cleaned,

CASE

  WHEN total\_delayed\_payments = 0 THEN 1

  WHEN total\_delayed\_payments <= 3 THEN 2

  WHEN total\_delayed\_payments <= 6 THEN 3

  WHEN total\_delayed\_payments <= 9 THEN 4

  WHEN total\_delayed\_payments >=10 THEN 5

END AS total\_delayed\_payments\_qtl,

CASE

  WHEN total\_delayed\_payments = 0 THEN '1. Sin retraso'

  WHEN total\_delayed\_payments <= 3 THEN '2. Leve'

  WHEN total\_delayed\_payments <= 6 THEN '3. Moderado'

  WHEN total\_delayed\_payments <= 9 THEN '4. Alto'

  WHEN total\_delayed\_payments >=10 THEN '5. Severo'

END AS total\_delayed\_payments\_category

FROM `banco-riesgo-relativo.datos\_banco.user\_info\_f` AS u

LEFT JOIN `banco-riesgo-relativo.datos\_banco.default` AS d

ON u.user\_id = d.user\_id

LEFT JOIN `banco-riesgo-relativo.datos\_banco.loans\_outstanding\_by\_user` AS lo

ON u.user\_id = lo.user\_id

LEFT JOIN `banco-riesgo-relativo.datos\_banco.loan\_detail\_c` AS ld

ON u.user\_id = ld.user\_id

WHERE ld.using\_lines\_not\_secured\_personal\_assets < 10000

      AND

      ld.debt\_ratio < 100000

      AND

      ld.total\_delayed\_payments < 250

ORDER BY u.user\_id desc;

**-- Ejemplo de vista con cálculo de Riesgo Relativo**

WITH tbl\_totales AS(

  SELECT

    salary\_qtl AS qtl,

    salary\_category as category\_name,

    MIN(last\_month\_salary\_cleaned) AS min\_value,

    MAX(last\_month\_salary\_cleaned) AS max\_value,

    ROUND(AVG(last\_month\_salary\_cleaned),2) AS avg\_value,

    SUM(CASE WHEN default\_flag = 1 THEN 1 ELSE 0 END) AS registros\_default\_1,

    SUM(CASE WHEN default\_flag = 0 THEN 1 ELSE 0 END) AS registros\_default\_0,

    COUNT(\*) as total\_registros,

  FROM `banco-riesgo-relativo.datos\_banco.user\_all\_data`

  GROUP BY salary\_qtl, salary\_category

),

Incidencia AS (

  SELECT s.\*,

    -- Calcula la tasa de incidencia

    Round(CAST(registros\_default\_1 AS FLOAT64) / total\_registros,4) AS tasa\_incidencia\_default\_1,

  FROM tbl\_totales AS s

),

IncidenciaCombinada AS (

  SELECT

    i.\*,

   ROUND( (SELECT SUM(registros\_default\_1) FROM tbl\_totales WHERE qtl != i.qtl) /

    (SELECT SUM (total\_registros) FROM tbl\_totales WHERE qtl != i.qtl),4) AS tasa\_incidencia\_combinada

  FROM Incidencia i

),

RR AS (

  SELECT ic.\*, ROUND(ic.tasa\_incidencia\_default\_1 / ic.tasa\_incidencia\_combinada,4) AS RR\_Qtl

  FROM IncidenciaCombinada ic

)

SELECT \* from RR

ORDER BY qtl;

**-- Union de todas las tablas de RR**

SELECT \*, 'age' AS columna FROM `datos\_banco.age\_rr`

UNION ALL

SELECT \*, 'debt\_ratio' FROM `datos\_banco.debt\_ratio\_rr`

UNION ALL

SELECT \*, 'salary' FROM `datos\_banco.salary\_rr`

UNION ALL

SELECT \*, 'total\_delayed\_payments' FROM `datos\_banco.total\_delayed\_payments\_rr`

UNION ALL

SELECT \*, 'total\_loans' FROM `datos\_banco.total\_loans\_rr`

UNION ALL

SELECT \*, 'using\_lines' FROM `datos\_banco.using\_lines\_rr`

UNION ALL

SELECT \*, 'more\_90\_days' FROM `datos\_banco.more\_90\_day\_rr`

UNION ALL

SELECT \*, 'dependents' FROM `datos\_banco.dependents\_rr`

order by columna, qtl