**Event source fill**  
df['EVENT\_SOURCE']=pd.Series(df['EVENT\_SOURCE']).astype(str).str.zfill(10)

df.head()

**Add date col**  
df['PREDICTION\_DATE'] = pd.to\_datetime('April-2024')

Add null cols  
  
REFERENCE\_NUM = ''

ESCALATED\_DATE = ''

ESCALATED\_REASON = ''

ACTION\_TAKEN = ''

CUST\_OBSERVATION\_FEEDBACK = ''

ESCALATED\_TO = ''

UPDATED\_DATE = ''

df = df.assign(

REFERENCE\_NUM = REFERENCE\_NUM,

ESCALATED\_DATE = ESCALATED\_DATE,

ESCALATED\_REASON = ESCALATED\_REASON,

ACTION\_TAKEN = ACTION\_TAKEN,

CUST\_OBSERVATION\_FEEDBACK = CUST\_OBSERVATION\_FEEDBACK,

ESCALATED\_TO = ESCALATED\_TO,

UPDATED\_DATE = UPDATED\_DATE)

Renale cols  
df = df.rename(columns= {'RANK':'CREDIT\_RANK'})

Df

Oracle  
  
connect db-create tbl

import cx\_Oracle as orcCon

from cx\_Oracle import DatabaseError

# Define Oracle database connection details

ip = '172.25.0.130'

port = 1521

SID = 'SLTDWH'

# dsn\_tns = cx\_Oracle.makedsn(ip, port, SID)

# Specify the table name

table\_name = 'CHURN\_PRED\_MONTLY\_REPORT'

# Define the connection details

username = 'SLT018970'

password = 'SLTTele##018970'

# Initialize cursor and connection variables

cursor = None

conn = None

try:

# Establish a connection to the Oracle database

conn = orcCon.connect(f'{username}/{password}@{ip}:{port}/{SID}')

if conn:

print("cx\_Oracle version:", orcCon.version)

print("Database version:", conn.version)

print("Client version:", orcCon.clientversion())

# Now execute the SQL query

cursor = conn.cursor()

print("You're connected.................")

# Drop table if exists

print(f'Dropping {table\_name} table if exists............')

cursor.execute(f"BEGIN EXECUTE IMMEDIATE 'DROP TABLE {table\_name}'; EXCEPTION WHEN OTHERS THEN NULL; END;")

# Creating the table with the new columns

print(f'Creating {table\_name} table............')

cursor.execute(f"""CREATE TABLE {table\_name} (

ACCOUNT\_NUM VARCHAR(20) NOT NULL,

BILLING\_CENTRE\_CODE VARCHAR(20),

EVENT\_SOURCE VARCHAR(20) NOT NULL,

CUSTOMER\_MOBILE VARCHAR(20),

LOYALITY VARCHAR(20),

RTOM\_CODE VARCHAR(20),

RTOM\_DESC VARCHAR(20),

CHURN\_PROB FLOAT NOT NULL,

AB\_STATUS VARCHAR(20) NOT NULL,

V\_PRODUCT\_STATUS VARCHAR(20),

BB\_PRODUCT\_STATUS VARCHAR(20),

PEO\_PRODUCT\_STATUS VARCHAR(20),

MEDIUM VARCHAR(20) NOT NULL,

CREDIT\_SCORE FLOAT NOT NULL,

CREDIT\_RANK INT NOT NULL,

AVG\_PAYMENT FLOAT NOT NULL,

SUBSCRIPTION VARCHAR(20) NOT NULL,

PREDICTION\_DATE VARCHAR(20) NOT NULL,

ACTION\_TAKEN VARCHAR(50),

CUST\_OBSERVATION\_FEEDBACK VARCHAR(50),

ESCALATED\_TO VARCHAR(50),

REFERENCE\_NUM VARCHAR(50),

ESCALATED\_DATE VARCHAR(50) ,

ESCALATED\_REASON VARCHAR(255),

UPDATED\_DATE VARCHAR(50)

)""")

print(f"{table\_name} table is created..............")

except DatabaseError as e:

err, = e.args

print("Oracle-Error-Code:", err.code)

print("Oracle-Error-Message:", err.message)

finally:

if cursor is not None:

# Close the cursor

cursor.close()

if conn is not None:

# Close the connection

conn.close()

insert

import cx\_Oracle as orcCon

from cx\_Oracle import DatabaseError

import pandas as pd # Assuming pandas is imported for DataFrame operations

try:

# Define Oracle database connection details

ip = '172.25.0.130'

port = 1521

SID = 'SLTDWH'

username\_b = 'SLT018970'

password\_b = 'SLTTele##018970'

# Establish a connection to the Oracle database

conn\_b = orcCon.connect(f'{username\_b}/{password\_b}@{ip}:{port}/{SID}')

if conn\_b:

print("cx\_Oracle version:", orcCon.version)

print("Database version:", conn\_b.version)

print("Client version:", orcCon.clientversion())

cursor\_b = conn\_b.cursor()

print("You're connected: ")

# Data Insertion

print('Inserting data into table....')

# Define a list to store values for bulk insertion

data\_values = [tuple(row) for \_, row in df.iterrows()]

# Define the SQL statement with bind variables for each column

sql = f"INSERT INTO CHURN\_PRED\_MONTLY\_REPORT(ACCOUNT\_NUM, BILLING\_CENTRE\_CODE, EVENT\_SOURCE, CUSTOMER\_MOBILE, LOYALITY, RTOM\_CODE, RTOM\_DESC, CHURN\_PROB, AB\_STATUS, V\_PRODUCT\_STATUS, BB\_PRODUCT\_STATUS, PEO\_PRODUCT\_STATUS, MEDIUM, CREDIT\_SCORE, CREDIT\_RANK, AVG\_PAYMENT, SUBSCRIPTION, PREDICTION\_DATE, ACTION\_TAKEN, CUST\_OBSERVATION\_FEEDBACK, ESCALATED\_TO, REFERENCE\_NUM, ESCALATED\_DATE, ESCALATED\_REASON,UPDATED\_DATE) VALUES (:1, :2, :3, :4, :5, :6, :7, :8, :9, :10, :11, :12, :13, :14, :15, :16, :17, :18, :19, :20, :21, :22, :23, :24, :25)"

# Execute the insert statement with executemany

try:

cursor\_b.executemany(sql, data\_values)

except Exception as e:

for index, row in enumerate(data\_values):

try:

cursor\_b.execute(sql, row) # Attempt to execute the statement for each row individually

except Exception as inner\_e:

print(f"Error occurred for row {index + 1}: {inner\_e}")

print(f"Problematic data row: {row}")

print("Error occurred during insertion:", e)

raise # Reraise the exception to propagate it further

# Commit the transaction

conn\_b.commit()

print("Record inserted successfully")

except DatabaseError as e:

err, = e.args

print("Oracle-Error-Code:", err.code)

print("Oracle-Error-Message:", err.message)

finally:

if 'cursor\_b' in locals() and cursor\_b is not None:

cursor\_b.close()

if 'conn\_b' in locals() and conn\_b is not None:

conn\_b.close()

connect  
  
import cx\_Oracle as orcCon

from cx\_Oracle import DatabaseError

def get\_row\_column\_count(cursor):

#Fetch and return the number of rows and columns from the cursor description.

num\_rows = len(cursor.fetchall())

num\_columns = len(cursor.description)

return num\_rows, num\_columns

def main():

ip = '172.25.0.130'

port = 1521

SID = 'SLTDWH'

username\_b = 'SLT018970'

password\_b = 'SLTTele##018970'

table\_name = 'CHURN\_PRED\_MONTLY\_REPORT'

try:

conn\_b = orcCon.connect(f'{username\_b}/{password\_b}@{ip}:{port}/{SID}')

if conn\_b:

print("cx\_Oracle version:", orcCon.version)

print("Database version:", conn\_b.version)

print("Client version:", orcCon.clientversion())

print("You're connected: ")

cursor\_b = conn\_b.cursor()

cursor\_b.execute(f"SELECT \* FROM {table\_name}")

num\_rows, num\_columns = get\_row\_column\_count(cursor\_b)

print(f"Number of rows: {num\_rows}")

print(f"Number of columns: {num\_columns}")

except DatabaseError as e:

err, = e.args

print("Oracle-Error-Code:", err.code)

print("Oracle-Error-Message:", err.message)

finally:

if 'cursor\_b' in locals():

cursor\_b.close()

if 'conn\_b' in locals():

conn\_b.close()

if \_\_name\_\_ == "\_\_main\_\_":

main()  
  
  
  
check-retrieve

import cx\_Oracle as orcCon

from cx\_Oracle import DatabaseError

def get\_first\_and\_last\_rows(cursor):

# Fetch the first row

first\_row = cursor.fetchone()

# Fetch all rows into a list

cursor.execute(cursor.statement) # Re-execute the cursor query to ensure cursor is at the beginning

all\_rows = cursor.fetchall()

# Extract the last row

last\_row = all\_rows[-1] if all\_rows else None

return first\_row, last\_row

def main():

ip\_b = '172.25.0.130'

port\_b = 1521

SID\_b = 'SLTDWH'

username\_b = 'SLT018970'

password\_b = 'SLTTele##018970'

table\_name = 'CHURN\_PRED\_MONTLY\_REPORT'

try:

conn\_b = orcCon.connect(f'{username\_b}/{password\_b}@{ip\_b}:{port\_b}/{SID\_b}')

if conn\_b:

print("cx\_Oracle version:", orcCon.version)

print("Database version:", conn\_b.version)

print("Client version:", orcCon.clientversion())

print("You're connected: ")

cursor\_b = conn\_b.cursor()

cursor\_b.execute(f"SELECT \* FROM {table\_name}")

first\_row, last\_row = get\_first\_and\_last\_rows(cursor\_b)

# Print first and last rows

print("First row:", first\_row)

print("Last row:", last\_row)

except DatabaseError as e:

err, = e.args

print("Oracle-Error-Code:", err.code)

print("Oracle-Error-Message:", err.message)

finally:

cursor\_b.close()

conn\_b.close()

if \_\_name\_\_ == "\_\_main\_\_":

main()