Tema 2 – MDB

# **Exerciții oracledb:**

Ex 1: Să se returneze într-un *df* componenta, anul de fabricatie și pretul manoperei pentru autoturismele Ford și Jeep. Pe setul din *df*, majorați cu 10% pretul manoperei pentru autoturismele fabricate inainte de 2010 și care au componenta BATTERY defectă. Salvați modificările într-un fișier .csv.

import oracledb  
import pandas as pd  
  
  
dsn\_str = "193.226.34.57:1521/orclpdb.docker.internal"  
user = "MS\_DBA1"  
password = "oracle"  
  
  
connection = oracledb.connect(user=user, password=password, dsn=dsn\_str)  
  
  
cursor = connection.cursor()  
  
query = """  
SELECT componenta, an\_fabricatie, pret\_manopera  
FROM t\_clienti\_daune  
WHERE marca IN ('FORD', 'JEEP')  
"""  
  
  
cursor.execute(query)  
  
  
results = cursor.fetchall()  
  
  
df = pd.DataFrame(results, columns=["Componenta", "An\_Fabricatie", "Pret\_Manopera"])  
  
  
df["An\_Fabricatie"] = pd.to\_numeric(df["An\_Fabricatie"], errors="coerce")  
df["Pret\_Manopera"] = pd.to\_numeric(df["Pret\_Manopera"], errors="coerce")  
  
  
df["Componenta"] = df["Componenta"].astype(str).str.strip()  
  
  
df.loc[  
 (df["An\_Fabricatie"] < 2010) &  
 (df["Componenta"].str.upper() == "BATTERY"),  
 "Pret\_Manopera"  
] \*= 1.10  
  
  
df.to\_csv("output\_ford\_jeep.csv", index=False)  
print("Fișier salvat: output\_ford\_jeep.csv")  
  
  
cursor.close()  
connection.close()

A screenshot of a computer

AI-generated content may be incorrect.

Output:

A screenshot of a computer program

AI-generated content may be incorrect.

Ex2: Încărcați într-un *df* marca, modelul, valoarea medie și numărul de daune pe fiecare model și marcă. Afișați numărul de autoturisme pentru care valoarea medie depășește 400 lei. Reprezentați grafic modelele care au înregistrat mai mult de 200 de daune.

A screenshot of a computer

AI-generated content may be incorrect.

import oracledb  
import pandas as pd  
import matplotlib.pyplot as plt  
dsn\_str = "193.226.34.57:1521/orclpdb.docker.internal"  
user = "MS\_DBA1"  
password = "oracle"  
connection = oracledb.connect(user=user, password=password, dsn=dsn\_str)  
cursor = connection.cursor()  
query = """  
SELECT marca, model, AVG(valoare\_dauna) AS valoare\_medie, COUNT(\*) AS numar\_daune  
FROM t\_clienti\_daune  
GROUP BY marca, model  
"""  
cursor.execute(query)  
results = cursor.fetchall()  
df = pd.DataFrame(results, columns=["Marca", "Model", "Valoare\_Medie", "Numar\_Daune"])  
df["Valoare\_Medie"] = pd.to\_numeric(df["Valoare\_Medie"], errors="coerce")  
df["Numar\_Daune"] = pd.to\_numeric(df["Numar\_Daune"], errors="coerce")  
num\_masini\_400 = df[df["Valoare\_Medie"] > 400].shape[0]  
print(f"Numărul de autoturisme cu valoarea medie a daunelor peste 400 lei: {num\_masini\_400}")  
df\_filtrat = df[df["Numar\_Daune"] > 200]  
plt.figure(figsize=(12, 6))  
plt.barh(df\_filtrat["Model"], df\_filtrat["Numar\_Daune"], color="steelblue")  
plt.xlabel("Număr Daune")  
plt.ylabel("Model")  
plt.title("Modelele cu peste 200 de daune")  
plt.gca().invert\_yaxis()  
plt.show()  
cursor.close()  
connection.close()

Ex3: Încărcați într-un *df* numele, suma solicitată, suma din depozite și fidelitatea clienților cu vârsta > 30 de ani care au solicitat un credit mai mare de 10.000 lei. Verificați în *df* dacă suma din depozit este mai mare decât suma solicitată și pentru acești clienți modificați fidelitatea în 5 (doar în *df*).

A computer screen shot of a program

AI-generated content may be incorrect.

import oracledb  
import pandas as pd  
dsn\_str = "193.226.34.57:1521/orclpdb.docker.internal"  
user = "MS\_DBA1"  
password = "oracle"  
connection = oracledb.connect(user=user, password=password, dsn=dsn\_str)  
cursor = connection.cursor()  
query = """  
SELECT nume\_client,   
 NVL(suma\_solicitata, 0) AS suma\_solicitata,   
 NVL(suma\_depozit, 0) AS suma\_depozit,   
 NVL(fidelitate, 0) AS fidelitate  
FROM CLIENTI\_NOI  
WHERE varsta > 30 AND NVL(suma\_solicitata, 0) > 10000  
"""  
cursor.execute(query)  
results = cursor.fetchall()  
df = pd.DataFrame(results, columns=["NUME\_CLIENT", "SUMA\_SOLICITATA", "SUMA\_DEPOZIT", "FIDELITATE"])  
df["SUMA\_SOLICITATA"] = pd.to\_numeric(df["SUMA\_SOLICITATA"], errors="coerce").fillna(0)  
df["SUMA\_DEPOZIT"] = pd.to\_numeric(df["SUMA\_DEPOZIT"], errors="coerce").fillna(0)  
df["FIDELITATE"] = pd.to\_numeric(df["FIDELITATE"], errors="coerce").fillna(0).astype(int)  
df.loc[df["SUMA\_DEPOZIT"] > df["SUMA\_SOLICITATA"], "FIDELITATE"] = 5  
print(df)  
df.to\_csv("output\_fidelitate.csv", index=False)  
print("Fișier salvat: output\_fidelitate.csv")  
cursor.close()  
connection.close()

Ex4: Încărcați într-un *df*  profesia, venitul anual, suma din depozite și suma solicitată *pe fiecare profesie*. În *df* adăugați o nouă coloană pentru a calcula gradul de îndatorare *pe fiecare profesie* (suma\_solicitata/(venit\_anual+suma\_depozit)\*100).

A screenshot of a computer program

AI-generated content may be incorrect.

import oracledb  
import pandas as pd  
dsn\_str = "193.226.34.57:1521/orclpdb.docker.internal"  
user = "MS\_DBA1"  
password = "oracle"  
  
connection = oracledb.connect(user=user, password=password, dsn=dsn\_str)  
cursor = connection.cursor()  
query = """  
SELECT profesia, venit\_anual, suma\_depozit, suma\_solicitata  
FROM CLIENTI\_NOI  
"""  
cursor.execute(query)  
results = cursor.fetchall()  
  
df = pd.DataFrame(results, columns=["Profesia", "Venit\_Anual", "Suma\_Depozit", "Suma\_Solicitata"])  
  
df["Venit\_Anual"] = pd.to\_numeric(df["Venit\_Anual"], errors="coerce")  
df["Suma\_Depozit"] = pd.to\_numeric(df["Suma\_Depozit"], errors="coerce")  
df["Suma\_Solicitata"] = pd.to\_numeric(df["Suma\_Solicitata"], errors="coerce")  
  
df["Grad\_Indatorare"] = (df["Suma\_Solicitata"] / (df["Venit\_Anual"] + df["Suma\_Depozit"])) \* 100  
  
print(df)  
  
df.to\_csv("grad\_indatorare.csv", index=False)  
print("Fișier salvat: grad\_indatorare.csv")  
  
cursor.close()  
connection.close()

Ex5: Încărcați într-un *df* starea civilă, profesia și suma totală solicitată grupată în funcție de aceste atribute. Introduceți de la tastatură profesia și pentru aceasta reprezentați grafic suma solicitată în funcție de starea civilă.

A screenshot of a computer program

AI-generated content may be incorrect.

import oracledb  
import pandas as pd  
import matplotlib.pyplot as plt  
  
# Configurare conexiune Oracle  
dsn\_str = "193.226.34.57:1521/orclpdb.docker.internal"  
user = "MS\_DBA1"  
password = "oracle"  
  
# Creare conexiune  
connection = oracledb.connect(user=user, password=password, dsn=dsn\_str)  
cursor = connection.cursor()  
  
# Interogare SQL pentru extragerea datelor necesare  
query = """  
SELECT stare\_civila, profesia, SUM(suma\_solicitata) AS suma\_totala  
FROM CLIENTI\_NOI  
GROUP BY stare\_civila, profesia  
"""  
  
# Executăm interogarea  
cursor.execute(query)  
  
# Preluăm datele într-o listă  
results = cursor.fetchall()  
  
# Creare DataFrame cu numele coloanelor  
df = pd.DataFrame(results, columns=["Stare\_Civila", "Profesia", "Suma\_Totala"])  
  
# Conversie date la tip numeric  
df["Suma\_Totala"] = pd.to\_numeric(df["Suma\_Totala"], errors="coerce")  
  
# Afișare rezultate  
print(df)  
  
# Salvare în fișier CSV  
df.to\_csv("suma\_solicitata\_pe\_profesie.csv", index=False)  
print("Fișier salvat: suma\_solicitata\_pe\_profesie.csv")  
  
# Introducere profesie de la tastatură  
profesie\_input = input("Introduceți profesia pentru care doriți să vizualizați suma solicitată: ")  
  
# Filtrare după profesie  
df\_filtrat = df[df["Profesia"].str.lower() == profesie\_input.lower()]  
  
# Verificare dacă există date pentru profesia introdusă  
if df\_filtrat.empty:  
 print(f"Nu există date pentru profesia: {profesie\_input}")  
else:  
 # Reprezentare grafică  
 plt.figure(figsize=(10, 6))  
 plt.bar(df\_filtrat["Stare\_Civila"], df\_filtrat["Suma\_Totala"], color="skyblue", edgecolor="black")  
 plt.xlabel("Stare Civilă")  
 plt.ylabel("Suma Totală Solicitată")  
 plt.title(f"Suma solicitată în funcție de starea civilă pentru profesia {profesie\_input}")  
 plt.xticks(rotation=45)  
 plt.grid(axis="y", linestyle="--", alpha=0.7)  
 plt.show()  
  
# Închidere conexiune  
cursor.close()  
connection.close()

**Teme:**

1. Utilizând SQL Alchemy, oracledb si pandas importati datele dintr-un fișier .csv la alegere, salvați datele in BD Oracle in contul individual si realizați cel puțin 10 interogări si prelucrări asupra tabelei.

import pandas as pd  
import oracledb  
from sqlalchemy import create\_engine  
from sqlalchemy.types import Integer, String, Float  
db\_user = "NAFORNITAD\_30"  
db\_password = "STUD"  
hostname = "193.226.34.57"  
port = "1521"  
service\_name = "orclpdb.docker.internal"  
dsn = f"(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST={hostname})(PORT={port}))"  
dsn += f"(CONNECT\_DATA=(SERVER=DEDICATED)(SERVICE\_NAME={service\_name})))"  
try:  
 connection = oracledb.connect(user=db\_user, password=db\_password, dsn=dsn)  
 print("✅ Conectare reușită la Oracle!")  
except Exception as e:  
 print(f"❌ Eroare la conectare: {e}")  
 exit()  
engine = create\_engine(  
 f"oracle+oracledb://{db\_user}:{db\_password}@{dsn}",  
 pool\_size=5,  
 max\_overflow=2  
)  
cursor = connection.cursor()  
query1 = "SELECT COUNT(\*) AS NUMAR\_CLIENTI FROM CLIENTI\_CREDITE"  
df1 = pd.read\_sql(query1, engine)  
print(df1)  
query2 = "SELECT AVG(SUMA\_SOLICITATA) AS SUMA\_MEDIE\_CREDITE FROM CLIENTI\_CREDITE"  
df2 = pd.read\_sql(query2, engine)  
print(df2)  
query3 = "SELECT COUNT(\*) AS NUMAR\_CLIENTI FROM CLIENTI\_CREDITE WHERE SUMA\_SOLICITATA > 10000"  
df3 = pd.read\_sql(query3, engine)  
print(df3)  
query4 = "SELECT NUME\_CLIENT, SUMA\_SOLICITATA, SUMA\_DEPOZIT FROM CLIENTI\_CREDITE WHERE SUMA\_DEPOZIT > SUMA\_SOLICITATA"  
df4 = pd.read\_sql(query4, engine)  
print(df4)  
query5 = """  
SELECT   
 CASE   
 WHEN VARSTA < 30 THEN 'Sub 30 ani'  
 WHEN VARSTA BETWEEN 30 AND 50 THEN '30-50 ani'  
 ELSE 'Peste 50 ani'  
 END AS INTERVAL\_VARSTA,  
 COUNT(\*) AS NUMAR\_CLIENTI  
FROM CLIENTI\_CREDITE  
GROUP BY   
 CASE   
 WHEN VARSTA < 30 THEN 'Sub 30 ani'  
 WHEN VARSTA BETWEEN 30 AND 50 THEN '30-50 ani'  
 ELSE 'Peste 50 ani'  
 END  
"""  
df5 = pd.read\_sql(query5, engine)  
print(df5)  
query6 = """  
SELECT NUME\_CLIENT, SUMA\_SOLICITATA   
FROM CLIENTI\_CREDITE   
ORDER BY SUMA\_SOLICITATA DESC   
FETCH FIRST 5 ROWS ONLY  
"""  
df6 = pd.read\_sql(query6, engine)  
print(df6)  
query7 = """  
SELECT FIDELITATE, SUM(SUMA\_SOLICITATA) AS TOTAL\_CREDITE  
FROM CLIENTI\_CREDITE  
GROUP BY FIDELITATE  
"""  
df7 = pd.read\_sql(query7, engine)  
print(df7)  
update\_query = """  
UPDATE CLIENTI\_CREDITE   
SET FIDELITATE = FIDELITATE + 1   
WHERE SUMA\_DEPOZIT > SUMA\_SOLICITATA  
"""  
cursor.execute(update\_query)  
connection.commit()  
print("✅ Fidelitatea a fost actualizată pentru clienții eligibili!")  
delete\_query = """  
DELETE FROM CLIENTI\_CREDITE   
WHERE SUMA\_SOLICITATA = 0 AND SUMA\_DEPOZIT = 0  
"""  
cursor.execute(delete\_query)  
connection.commit()  
print("✅ Clienții fără credite și depozite au fost șterși!")  
query10 = """  
SELECT NUME\_CLIENT, VARSTA, SUMA\_SOLICITATA   
FROM CLIENTI\_CREDITE   
WHERE VARSTA > 40 AND SUMA\_SOLICITATA > 20000  
"""  
df10 = pd.read\_sql(query10, engine)  
print(df10)  
cursor.close()  
connection.close()  
engine.dispose()  
print("✅ Conexiunea la baza de date a fost închisă.")

A screenshot of a computer program

AI-generated content may be incorrect.

1. Studiați modalitatea de aplicare a altor operatori sau funcții SQL in SQL Alchemy si aplicați aceste operații pe tabela creata la pasul 1.

import pandas as pd  
import oracledb  
from sqlalchemy import create\_engine  
db\_user = "NAFORNITAD\_30"  
db\_password = "STUD"  
hostname = "193.226.34.57"  
port = "1521"  
service\_name = "orclpdb.docker.internal"  
dsn = f"(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST={hostname})(PORT={port}))"  
dsn += f"(CONNECT\_DATA=(SERVER=DEDICATED)(SERVICE\_NAME={service\_name})))"  
engine = create\_engine(f"oracle+oracledb://{db\_user}:{db\_password}@{dsn}")  
query1 = "SELECT ROUND(AVG(SUMA\_SOLICITATA), 2) AS SUMA\_MEDIE\_ROTUNJITA FROM CLIENTI\_CREDITE"  
df1 = pd.read\_sql(query1, engine)  
print(df1)  
query2 = "SELECT COUNT(DISTINCT NUME\_CLIENT) AS CLIENTI\_UNICI FROM CLIENTI\_CREDITE"  
df2 = pd.read\_sql(query2, engine)  
print(df2)  
query3 = "SELECT NUME\_CLIENT, LENGTH(NUME\_CLIENT) AS LUNGIME\_NUME FROM CLIENTI\_CREDITE"  
df3 = pd.read\_sql(query3, engine)  
print(df3)  
query4 = "SELECT NUME\_CLIENT, UPPER(NUME\_CLIENT) AS NUME\_MARE, LOWER(NUME\_CLIENT) AS NUME\_MIC FROM CLIENTI\_CREDITE"  
df4 = pd.read\_sql(query4, engine)  
print(df4)  
query5 = "SELECT \* FROM CLIENTI\_CREDITE WHERE NUME\_CLIENT LIKE 'A%'"  
df5 = pd.read\_sql(query5, engine)  
print(df5)  
query6 = "SELECT NUME\_CLIENT, SUBSTR(NUME\_CLIENT, 1, 3) AS INITIALA FROM CLIENTI\_CREDITE"  
df6 = pd.read\_sql(query6, engine)  
print(df6)  
query7 = "SELECT NUME\_CLIENT, NVL(SUMA\_DEPOZIT, 0) AS SUMA\_DEPOZIT\_FARA\_NULL FROM CLIENTI\_CREDITE"  
df7 = pd.read\_sql(query7, engine)  
print(df7)  
query8 = "SELECT \* FROM CLIENTI\_CREDITE WHERE VARSTA BETWEEN 30 AND 50"  
df8 = pd.read\_sql(query8, engine)  
print(df8)  
query9 = """  
SELECT NUME\_CLIENT, SUMA\_SOLICITATA,  
 CASE   
 WHEN SUMA\_SOLICITATA < 5000 THEN 'Credit mic'  
 WHEN SUMA\_SOLICITATA BETWEEN 5000 AND 15000 THEN 'Credit mediu'  
 ELSE 'Credit mare'  
 END AS TIP\_CREDIT  
FROM CLIENTI\_CREDITE  
"""  
df9 = pd.read\_sql(query9, engine)  
print(df9)  
query10 = """  
SELECT NUME\_CLIENT, VARSTA,   
 EXTRACT(YEAR FROM SYSDATE) - VARSTA AS AN\_NASTERE  
FROM CLIENTI\_CREDITE  
"""  
df10 = pd.read\_sql(query10, engine)  
print(df10)  
engine.dispose()  
print("✅ Conexiunea la baza de date a fost închisă.")

A screenshot of a computer

AI-generated content may be incorrect.

1. Realizați cel puțin 3 operații de actualizare pe tabela creata utilizând SQL Alchemy sau oracledb
2. import oracledb  
   from sqlalchemy import create\_engine, text  
   db\_user = "NAFORNITAD\_30"  
   db\_password = "STUD"  
   hostname = "193.226.34.57"  
   port = "1521"  
   service\_name = "orclpdb.docker.internal"  
   dsn = f"(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST={hostname})(PORT={port}))"  
   dsn += f"(CONNECT\_DATA=(SERVER=DEDICATED)(SERVICE\_NAME={service\_name})))"  
   engine = create\_engine(f"oracle+oracledb://{db\_user}:{db\_password}@{dsn}")  
     
   try:  
    connection = oracledb.connect(user=db\_user, password=db\_password, dsn=dsn)  
    cursor = connection.cursor()  
    print("✅ Conectare reușită la Oracle!")  
    update\_query1 = """  
    UPDATE CLIENTI\_CREDITE   
    SET FIDELITATE = 5   
    WHERE SUMA\_DEPOZIT > SUMA\_SOLICITATA  
    """  
    cursor.execute(update\_query1)  
    connection.commit()  
    print("✅ Fidelitatea a fost actualizată pentru clienții eligibili!")  
    update\_query2 = """  
    UPDATE CLIENTI\_CREDITE   
    SET SUMA\_SOLICITATA = SUMA\_SOLICITATA \* 1.1  
    WHERE VARSTA > 30  
    """  
    cursor.execute(update\_query2)  
    connection.commit()  
    print("✅ Sumele solicitate au fost majorate cu 10% pentru clienții peste 30 de ani!")  
     
    print("✅ Sumele solicitate au fost majorate cu 10% pentru clienții fideli!")  
    update\_query3 = """  
    UPDATE CLIENTI\_CREDITE   
    SET SUMA\_SOLICITATA = SUMA\_SOLICITATA \* 0.8  
    WHERE SUMA\_DEPOZIT IS NULL  
    """  
    cursor.execute(update\_query3)  
    connection.commit()  
    print("✅ Sumele solicitate au fost reduse cu 20% pentru clienții fără depozite!")  
     
   except Exception as e:  
    print(f"❌ Eroare la actualizare: {e}")  
     
   finally:  
    cursor.close()  
    connection.close()  
    engine.dispose()  
    print("✅ Conexiunea la baza de date a fost închisă.")

A screenshot of a computer program

AI-generated content may be incorrect.