UNIVERSITATEA TEHNICĂ „Gheorghe Asachi” din IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DOMENIUL: Calculatoare și tehnologia informației

LAN PARTY CHAMPIONSHIP

Tema de casa la disciplina
BAZE DE DATE

Student:
Spătaru Alexandru 1308A

Introducere

Aplicația este creată cu scopul de a ține gestiunea bazei de date a unui turneu de tip LAN PARTY. Baza de date cuprinde 4 tabele:

- PLAYERS
- SCORES
- TEAMS
- RESULTS

Prin aceasta aplicație putem organiza și controla cu ușurință un astfel de eveniment. Am implementat funcțiile uzuale pentru interogarea unei baze de date, acestea fiind: SELECT, INSERT, UPDATE, DELETE.

Instalare aplicații necesare

Pentru funcționarea corecta a aplicației va trebui sa instalam:

1. PyCharm 2023.2.5 sau o versiune mai veche
2. Extensia cx_Oracle si tkinter pentru Python
3. Oracle Instant Client

La realizarea aplicației am folosit pentru partea de FRONT-END limbajul Python și mediul de dezvoltare PyCharm 2023.2.5 iar pentru partea de BACK-END limbajul SQL și mediile de dezvoltare SQL Developer și SQL Developer Data Modeler.

Data Modeler

Relațiile dintre tabele:

- 1 la 1 între tabelul SCORES și PLAYERS (unui jucător îi corespunde un scor unic)
- 1 la 1 între tabelul RESULT și TEAMS (unei echipe îi corespunde un rezultat unic)
- 1 la n între tabelul PLAYERS și TEAMS (un jucător poate face parte dintr-o singura echipa)

Descrierea constrângerilor:

Tabela PLAYERS

Constrângeri de tip NOT NULL pentru câmpurile FIRST_NAME,NICKNAME
Constrângere de tip UNIQUE pentru câmpul CNP
Constrângere de tip PRIMARY KEY pentru PLAYER_ID
Constrângeri de tip FOREIGN KEY pentru SCORE_ID, TEAM_ID
Constrângere de tip CHECK pentru câmpul BIRTH_DATE

Tabela SCORES

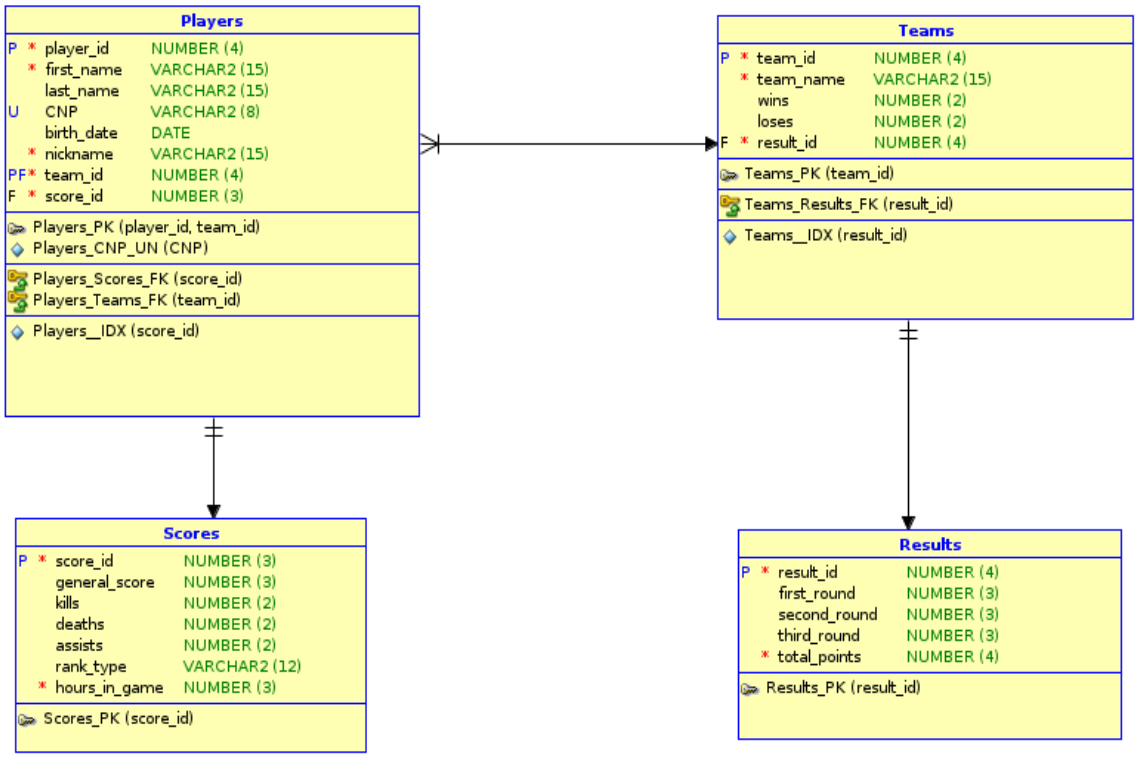
Constrângere de tip NOT NULL pentru câmpul HOURS_IN_GAME
Constrângere de tip PRIMARY KEY pentru SCORE_ID

Tabela TEAMS

Constrângere de tip PRIMARY KEY pentru TEAM_ID
Constrângere de tip NOT NULL pentru câmpul TEAM_NAME
Constrângere de tip FOREIGN KEY pentru RESULT_ID

Tabela RESULTS

Constrângere de tip PRIMARY KEY pentru RESULT_ID
Constrângere de tip NOT NULL pentru câmpul TOTAL_POINTS



Funcționalitatea aplicației

Funcția Select: Funcție ce permite vizualizarea datelor din tabele.

Sintaxa : SELECT (nume coloane / * pentru întreg tabelul) FROM (nume tabel)

Exemplu de utilizare a funcției SELECT pentru afișarea datelor în aplicația scrisă în Python:

```
def display_players():
    try:
        connection = connect_to_db()
        cursor = connection.cursor()

        cursor.execute('SELECT * FROM players ORDER BY PLAYER_ID')

        players = cursor.fetchall()

        cursor.close()
        connection.close()

        table_window = tk.Toplevel()
        table_window.title("Players Table")
```

Aceasta funcție se va apela deschiderea formularelor și la aplicarea operațiilor de modificare a datelor.

Rezultatul codului este :

Players Table

Index	PLAYER_ID	PRENUME	NUME	CNP	BIRTH_DATE	NICKNAME	TEAM_ID	SCORE_ID
1	1025	Matei	Catalin	50331017	2003-12-06 00:00:00	cata_001	4	6
2	1026	Luca	Ionel	50311073	2001-04-12 00:00:00	luca_ionel	5	8
3	1027	Bianca	Ana	60221902	2001-01-04 00:00:00	biaka	6	9
4	1028	Anamaria	Ursu	60403102	2004-09-02 00:00:00	ana_ursu99	4	11
5	1029	Sergiu	Ionescu	50001902	2000-08-21 00:00:00	sergiuFTW	6	12
6	1030	Cosmina	Tanase	70219323	2003-06-30 00:00:00	cosmina_t	4	13
7	1031	Florin	Diaconu	50774104	2000-02-29 00:00:00	Floriin_29	6	14
8	1032	Ana	Lungu	60143123	2000-01-01 00:00:00	ana2000	5	15
9	1033	Sabin	Filipescu	51234323	2003-12-31 00:00:00	sabinel99	5	16
10	1034	Viorel	Zaharia	50101077	2001-11-07 00:00:00	V2001	6	17

Pentru funcțiile INSERT, DELETE, UPDATE am realizat câte o funcție Python care va efectua comanda dorită la apăsarea butonului de pe interfața grafică corespunzător.

Codul:

Funcția insert:

```
def insert():
    try:
        connection = connect_to_db()
        if connection:
            cursor = connection.cursor()

            first_name = first_name_entry.get()
            last_name = last_name_entry.get()
            cnp = cnp_entry.get()
            birth_date = birth_date_entry.get()
            nickname = nickname_entry.get()
            team_id = int(team_list.get())
            score_id = int(scor_entry.get())

            cursor.execute("SELECT player_id_sequence.nextval FROM dual")
            next_id = cursor.fetchone()[0]

            sql = """
                INSERT INTO PLAYERS (PLAYER_ID, FIRST_NAME, LAST_NAME, CNP, BIRTH_DATE, NICKNAME, TEAM_ID, SCORE_ID)
                VALUES (:player_id, :first_name, :last_name, :cnp, TO_DATE(:birth_date, 'MM/DD/YYYY'), :nickname, :team_id, :score_id)
            """
            cursor.execute(sql, {
                'player_id': next_id,
                'first_name': first_name,
                'last_name': last_name,
                'cnp': cnp,
                'birth_date': birth_date,
                'nickname': nickname,
                'team_id': team_id,
                'score_id': score_id
            })
            connection.commit()
            cursor.close()
            connection.close()
            print("Player inserted successfully!")
    except cx_Oracle.DatabaseError as e:
        print(f"Error: {e}")
```

Funcția delete:

```
def delete():
    try:
        connection = connect_to_db()
        if connection:
            cursor = connection.cursor()
            player_id = id_player_entry.get()
            cursor.execute("DELETE FROM PLAYERS WHERE PLAYER_ID = :player_id", {'player_id': player_id})
            connection.commit()
            cursor.close()
            connection.close()
            print("Player deleted successfully!")
    except cx_Oracle.DatabaseError as e:
        print(f"Error: {e}")
```

Funcția Insert: Funcție ce permite adăugarea unei noi înregistrări în tabele.
Sintaxa : INSERT INTO (nume tabel) (specificare ordine coloane) VALUES (date introduse).
Exemplu de utilizare a funcției INSERT pentru adăugarea unei noi înregistrări în tabela SCORES:

```
sql = """
    INSERT INTO SCORES (SCORE_ID, GENERAL_SCORE, KILLS, DEATHS, ASSISTS, RANK_TYPE, HOURS_IN_GAME)
    VALUES (:score_id, :GENERAL_SCORE, :KILLS, :DEATHS, :ASSISTS, :RANK_TYPE, :HOURS_IN_GAME)
"""
```

Și rezultatul :

Scores Table

Index	SCORE_ID	GENERAL_SCORE	KILLS	DEATHS	ASSISTS	RANK_TYPE	HOURS_IN_GAME
1	6	255	98	43	59	Professional	231
2	8	700	99	25	84	Professional	987
3	9	345	22	21	6	Amateur	143
4	11	400	25	19	8	Professional	245
5	12	210	30	17	9	Semipro	189
6	13	999	97	14	32	Professional	949
7	14	342	17	17	10	Semipro	457
8	15	956	39	20	21	Professional	832
9	16	743	25	19	15	Semipro	700
10	17	100	10	27	4	Amateur	95

Funcția DELETE: Funcție ce permite eliminarea unei înregistrări din tabele.
Sintaxa : DELETE FROM (nume tabel) WHERE (condiție ex PLAYER_ID = 4)
Exemplu de utilizare a funcției DELETE pentru ștergerea unei înregistrări din tabela SCORES:

```
def delete():
    try:
        connection = connect_to_db()
        if connection:
            cursor = connection.cursor()
            score_id = id_score_entry.get()
            cursor.execute("DELETE FROM SCORES WHERE SCORE_ID = :score_id", {'score_id': score_id})
            connection.commit()
            cursor.close()
            connection.close()
            print("Score deleted successfully!")
    except cx_Oracle.DatabaseError as e:
        print(f"Error: {e}")
```

Rezultatul :

Scores Table

Index	SCORE_ID	GENERAL_SCORE	KILLS	DEATHS	ASSISTS	RANK_TYPE	HOURS_IN_GAME
1	6	255	98	43	59	Professional	231
2	8	700	99	25	84	Professional	987
3	9	345	22	21	6	Amateur	143
4	11	400	25	19	8	Professional	245
5	12	210	30	17	9	Semipro	189
6	13	999	97	14	32	Professional	949
7	14	342	17	17	10	Semipro	457
8	15	956	39	20	21	Professional	832
9	16	743	25	19	15	Semipro	700
10	17	100	10	27	4	Amateur	95

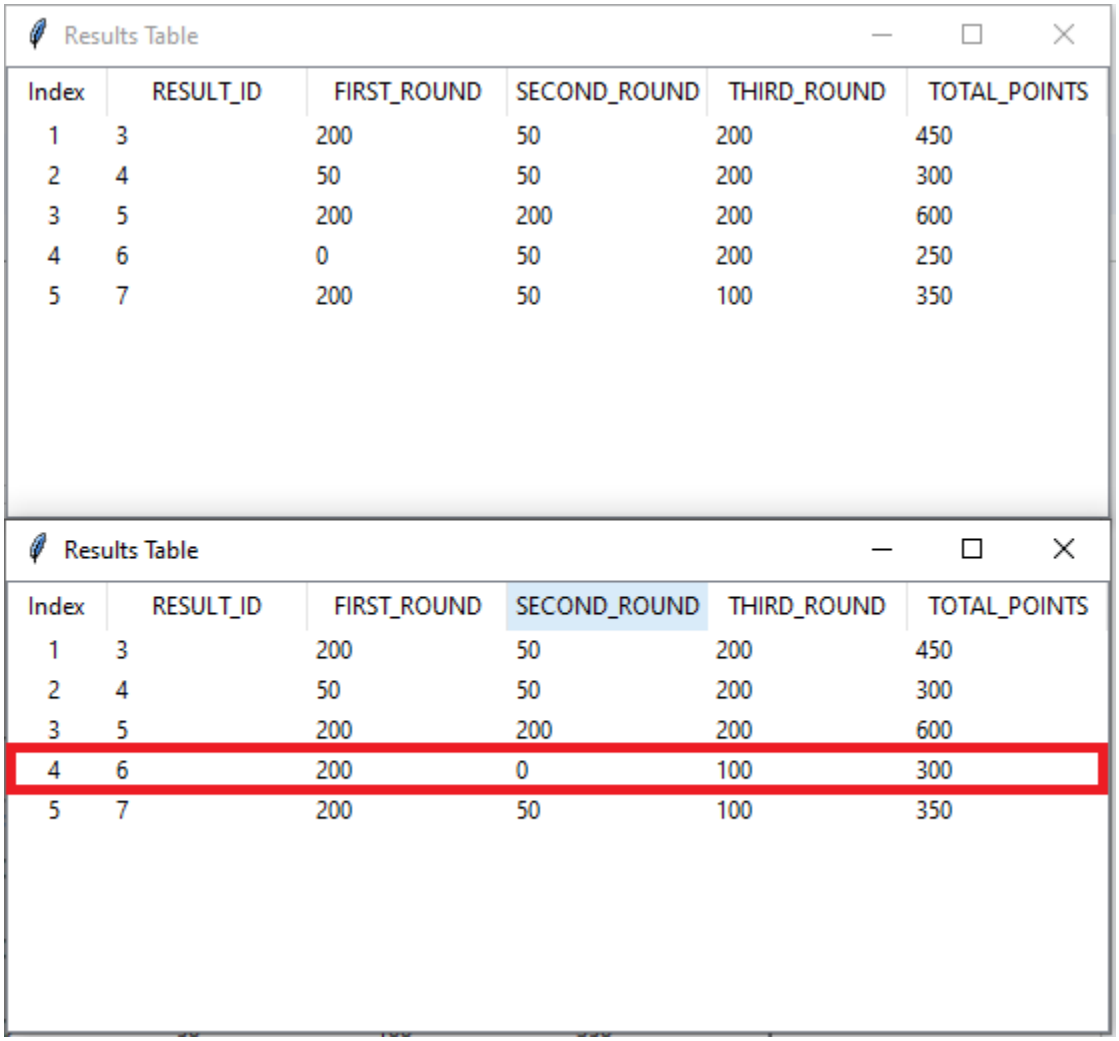
Scores Table

Index	SCORE_ID	GENERAL_SCORE	KILLS	DEATHS	ASSISTS	RANK_TYPE	HOURS_IN_GAME
1	6	255	98	43	59	Professional	231
2	8	700	99	25	84	Professional	987
3	9	345	22	21	6	Amateur	143
4	11	400	25	19	8	Professional	245
5	12	210	30	17	9	Semipro	189
6	13	999	97	14	32	Professional	949
7	14	342	17	17	10	Semipro	457
8	15	956	39	20	21	Professional	832
9	16	743	25	19	15	Semipro	700

Funcția UPDATE: Funcție ce permite actualizarea unei înregistrări dintr-un tabel.
Sintaxa : UPDATE (nume tabel) SET (nume coloana = valoare nouă...) WHERE (condiție)
Exemplu de utilizare a funcției UPDATE pentru modificarea unei înregistrări din tabela RESULTS:

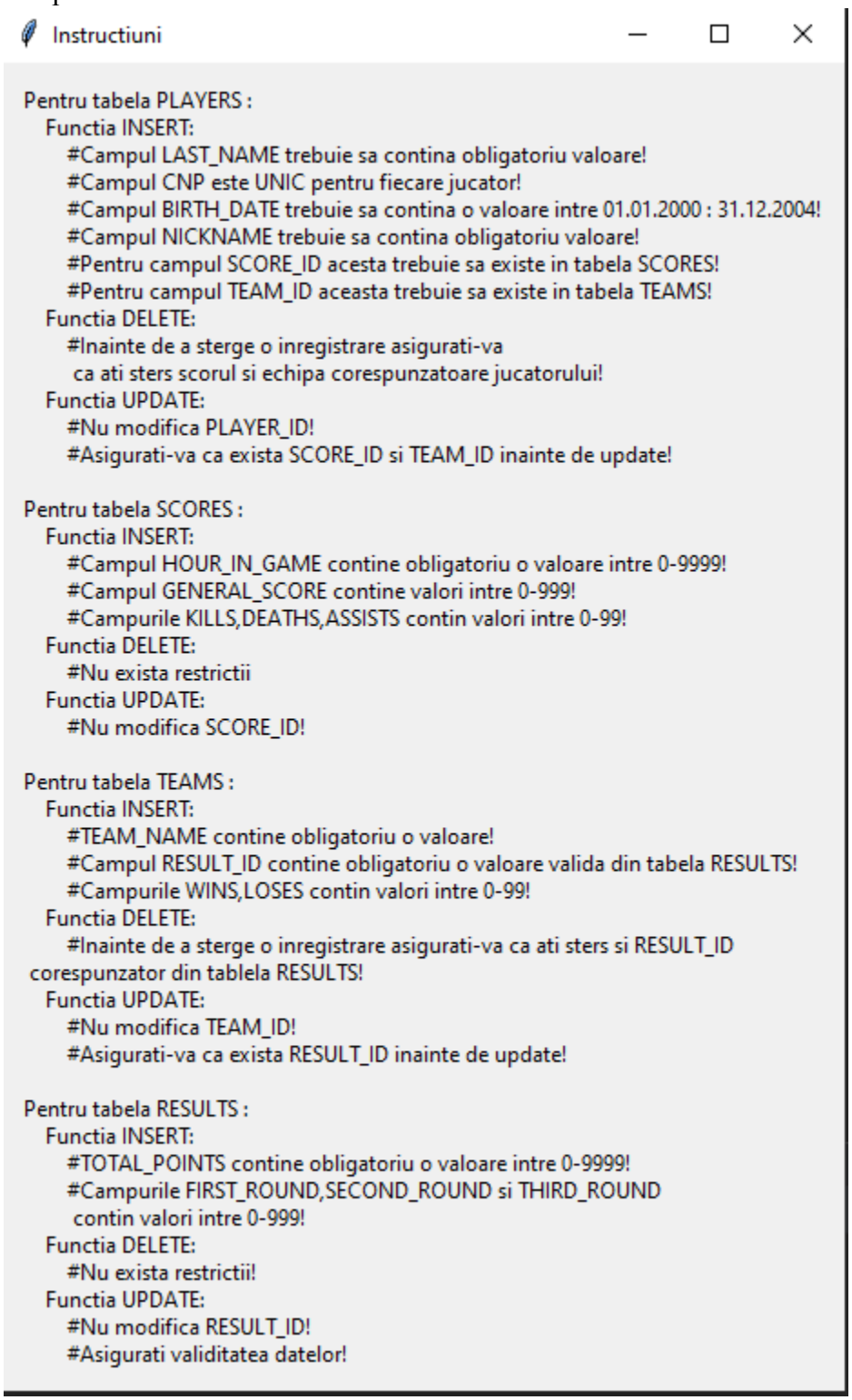
```
def update():
    try:
        connection = connect_to_db()
        if connection:
            cursor = connection.cursor()

            result_id = int(id_entry.get())
            first_round_value = int(first_type_var.get())
            second_round_value = int(second_type_var.get())
            third_round_value = int(third_type_var.get())
            total_points_value = int(totalpoints_entry.get())

            update_query = """
                UPDATE RESULTS
                SET first_round = :first_round,
                    second_round = :second_round,
                    third_round = :third_round,
                    total_points = :total_points
                WHERE result_id = :result_id
            """

            cursor.execute(update_query, {
                'first_round': first_round_value,
                'second_round': second_round_value,
                'third_round': third_round_value,
                'total_points': total_points_value,
                'result_id': result_id,
            })

            connection.commit()
            cursor.close()
            connection.close()
            print("Data updated successfully!")

    except cx_Oracle.DatabaseError as e:
        print(f"Error: {e}")
```

Rezultatul :


Results Table
Index RESULT_ID FIRST_ROUND SECOND_ROUND THIRD_ROUND TOTAL_POINTS
1 3 200 50 200 450
2 4 50 50 200 300
3 5 200 200 200 600
4 6 0 50 200 250
5 7 200 50 100 350
Results Table
Index RESULT_ID FIRST_ROUND SECOND_ROUND THIRD_ROUND TOTAL_POINTS
1 3 200 50 200 450
2 4 50 50 200 300
3 5 200 200 200 600
4 6 200 0 100 300
5 7 200 50 100 350

Instrucțiuni de folosire

Pentru o bună funcționare și informarea utilizatorilor am atașat și un formular de instrucțiuni pentru modificarea datelor din tabele.



Conectarea la baza de date se realizează prin următoarea secvență de cod :

```python
def connect_to_db():
    try:
        connection = cx_Oracle.connect('bd072', 'Alexandru2oo2', 'bd-dc.cs.tuiasi.ro:1539/orcl')
        return connection
    except cx_Oracle.DatabaseError as e:
        print(f"Error: {e}")
        return None

1 usage
def close_connection():
    global connection
    if connection:
        connection.close()
        print("Conexiunea la baza de date a fost închisă cu succes.")
    else:
        print("Nu există nicio conexiune la baza de date de închis.")
```

```python
try:
    connection = cx_Oracle.connect('bd072', 'Alexandru2oo2', 'bd-dc.cs.tuiasi.ro:1539/orcl')
    print("Conexiunea la baza de date s-a realizat cu succes.")
except cx_Oracle.DatabaseError as e:
    print(f"Eroare la conexiunea la baza de date: {e}")

root.protocol( name: "WM_DELETE_WINDOW", close_connection)
```