UNIVERSITATEA TEHNICĂ „GHEORGHE ASACHI” IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
SPECIALIZAREA CALCULATOARE SI TEHNOLOGIA INFORMAȚIEI
DISCIPLINA BAZE DE DATE

Seară de Film

Prof. Coordonator: Cătălin Mironeanu

Student: Martin Maria

Grupa: 1309A

1. Scopul

Aplicația a fost creată cu scopul gestionării unei baze de date a unor filme, ținând evidența filmelor preferate sau a celor pe care utilizatorul dorește să le vizioneze.

Baza de date cuprinde 4 tabele:

- Filme: fiecărui film îi va fi asignat un actor și un regizor deja existent in baza de date;
- Actori: se vor păstra informații despre actori;
- Regizori: se vor păstra informații despre regizori;
- Favorite: se vor insera doar filme din tabela "*Filme*", acestea nu pot fi duplicate.

Prin această aplicație am implementat funcționalitatea principalelor funcții de interogare a unei baze de date, putând, astfel, accesa o baza de date datorită bibliotecii "*oracledb*" din Python. Principalele funcții folosite din *SQL* sunt:

- SELECT
- INSERT
- UPDATE
- DELETE

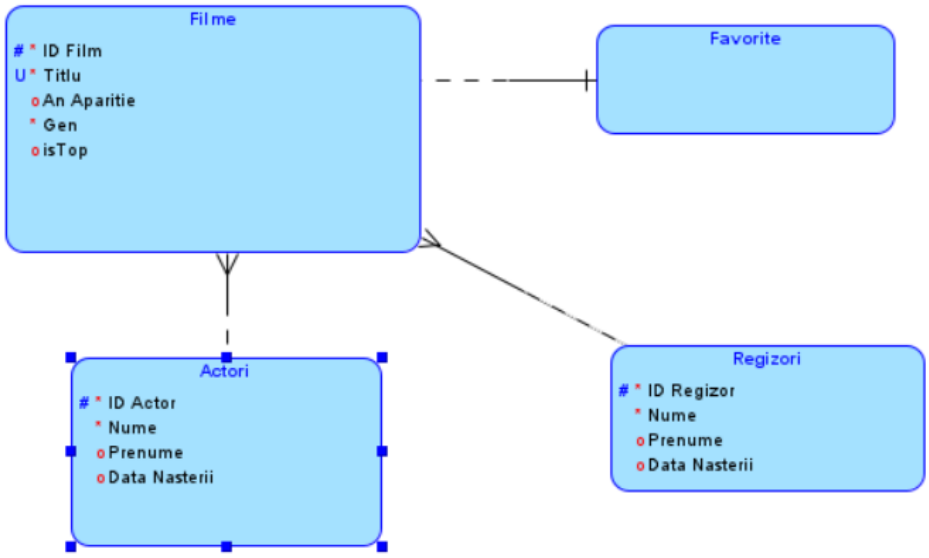
2. Tehnologii Folosite

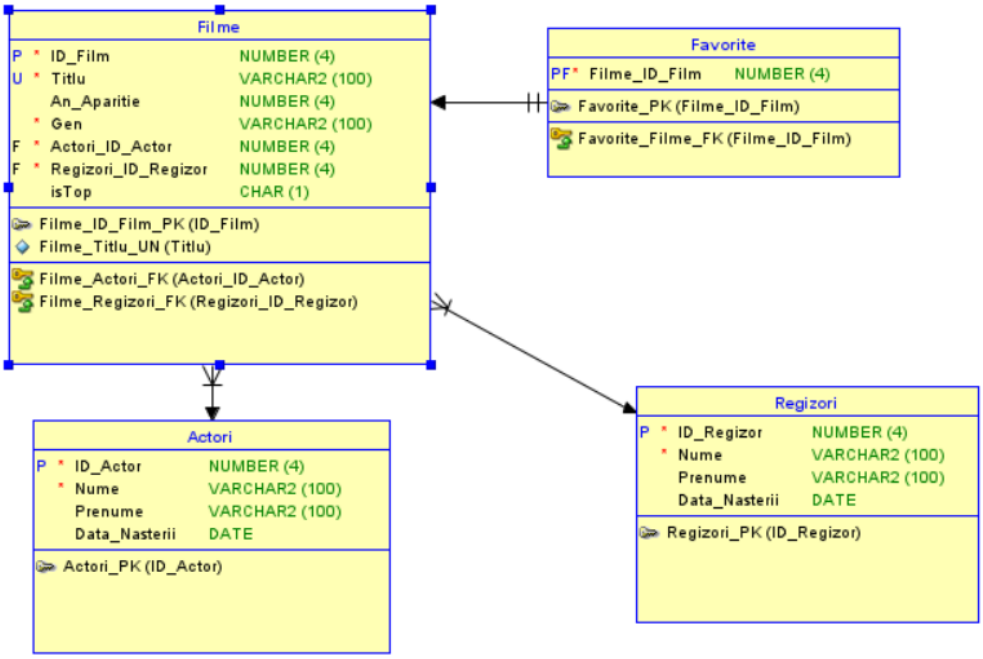
Pentru această aplicație am folosit limbajul *Python 3.9* cu următoarele librării:

- *sys*: Acest modul oferă acces la unele variabile utilizate sau întreținute de interpret și la funcții care interacționează puternic cu interpretul.
- *oracledb*: Driverul python-oracledb este un modul de extensie a limbajului de programare Python care permite programelor Python să se conecteze la baza de date Oracle. Python-oracledb este noul nume pentru popularul driver Oracle cx_Oracle.
- *PyQt5*: PyQt5 este un set cuprinzător de legături Python pentru Qt v5. Este implementat ca mai mult de 35 de module de extensie și permite ca Python să fie utilizat ca limbaj alternativ de dezvoltare a aplicațiilor la C++ pe toate platformele acceptate.
- *datetime:* Modulul datetime furnizează clase pentru manipularea datelor și orelor.

Baza de date a fost creată în *DataModeler* și este scrisă în limbajul *SQL*.

3. Diagrama logică și relațională





Baza de date descrisă are în componenta sa relații de tipul *1:n, 1:1.*

Între tabelele *Actori* și *Filme* remarcăm o relație de tipul *1:n*, întrucât un actor poate juca în mai multe filme.

Între tabelele *Regizori* și *Filme* remarcăm o relație de tipul *1:n*, întrucât un regizor poate face mai multe filme.

Între tabelele *Favorite* și *Filme* remarcăm o relație de tipul *1:1*, întrucât un film poate fi adăugat în lista de favorite o singura dată.

4. Constrângeri utilizate

De tip check:

Se verifică dacă lungimea numelui este de minim 2 caractere.

- *Nume Actor*
- *Nume Regizor*

De tip Unique:

- *Titlu Film:* nu pot exista filme cu același titlu în această bază de date.

De tip NOT NULL:

- *Titlu Film:* orice film trebuie sa aibă un titlu identificator;
- *Gen Film:* pentru a clasifica filmele este necesară existența unui gen;
- *Nume Actor:* este nevoie pentru a putea identifica date despre actor;
- *Nume Regizor:* este nevoie pentru a putea identifica date despre regizor.

De tip FOREIGN KEY:

Cu ajutorul acestor *FK* ne asigurăm că nu pot exista în tabela *Filme* actori sau regizori ce nu se află în baza de date.

- *Actori_ID_Actori*
- *Regizori_ID_Actori*

Cu acest *FK* ne asigurăm că putem adăuga în tabela *Favorite* doare filme care există în baza noastră de date.

- *Filme_ID_Film*

De tip PRIMARY KEY:

Nu pot fi mai mici decât 0 și sunt create cu autoincrement.

- *ID_Film*
- *ID_Actor*
- *ID_Regizor*
- *Filme_ID_Film*

5. Conectarea la baza de date

Conectarea la baza de date se realizează cu ajutorul librăriei Python *oracledb.*

```
oracledb.init_oracle_client()

connection = oracledb.connect(
    user="maria",
    password="2008Mm1206D!",
    dsn="localhost:1521/xe")

print("Successfully connected to Oracle Database")
```

6. Funcționaliatatea aplicației

Prin această aplicație am implementat vizual funcționalitatea principalelor funcții de interogare a unei baze de date, putând modifica sau accesa baza de date prin intermediul bibliotecilor Python menționate.

FUNCȚIA SELECT:

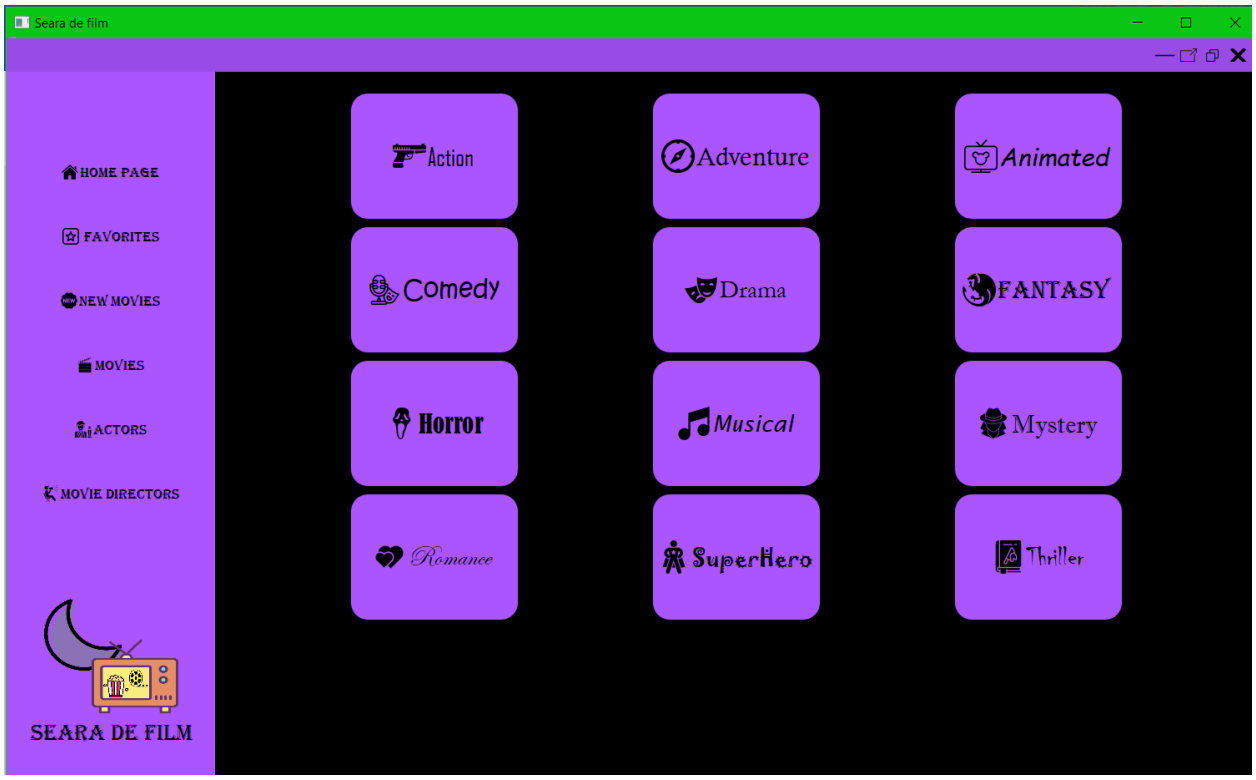
Este comanda folosită pentru obținerea datelor din baza de date.

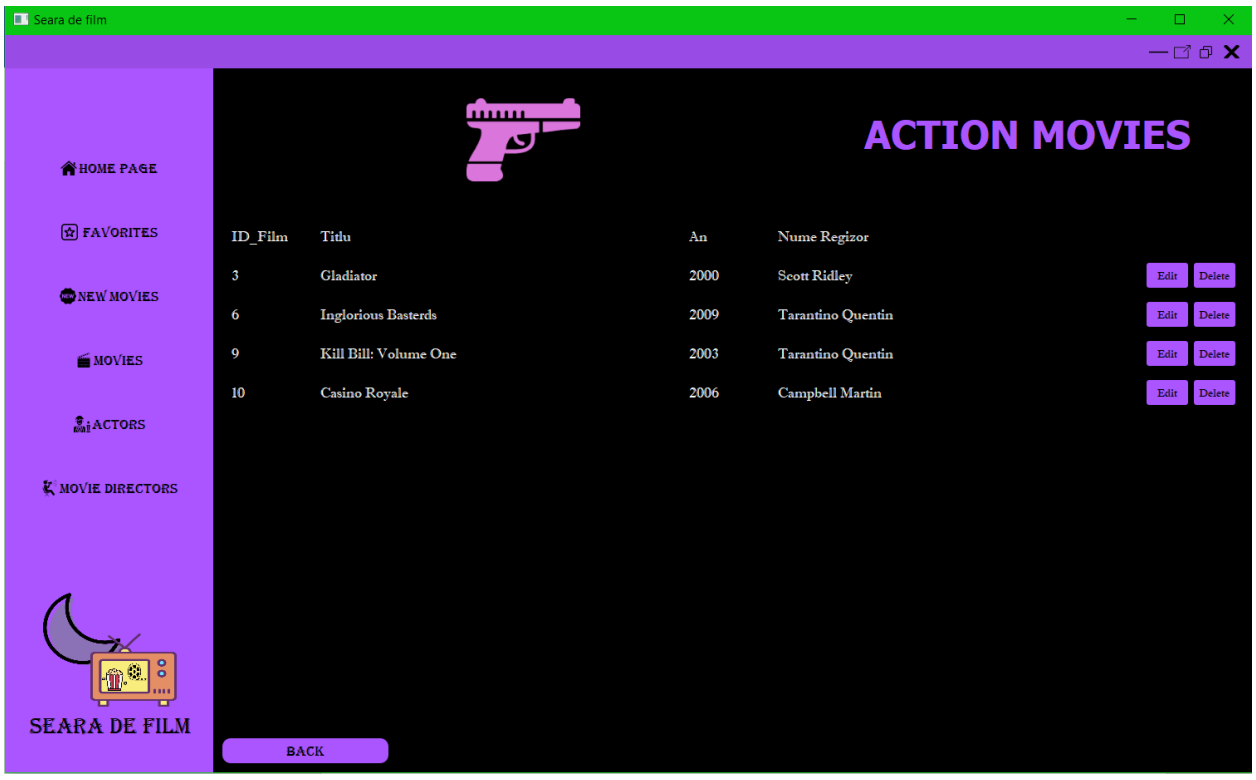
Exemplu pentru a afișa cele mai vizionate filme in anii 2000:

```
def init_home(self):
    cursor.execute(
        "SELECT (r.nume||' '||r.prenume), f.an_aparitie, f.titlu " +
        "FROM Filme f, Regizori r " +
        "WHERE isTop = 1 AND r.ID_Regizor=f.Regizori_ID_Regizor"
    )
    topVLayout = QVBoxLayout() #Layout-uri cu fiecare valoare
    topVLayout.addLayout(self.makeRow(False, False, 'title', [-1, 100, 0], "Nume Regizor", "An", "Titlu"))
    for text in cursor:
        topVLayout.addLayout(self.makeRow(False, False, 'home', [-1, 100, 0], *text))
    topVLayout.addStretch(1) #Spatiere
    self.topWidgetHome.setLayout(topVLayout) #Setam in Scroll area
```

Exemplu pentru clasificarea filmelor în funcție de gen:

```
def refresh_genrePage(self, genre: str):
    self.containerByGenre[genre].deleteLater()
    cursor.execute(
        "SELECT f.ID_Film, f.titlu, f.an_aparitie, (r.nume||' '||r.prenume) " +
        "FROM Filme f, Regizori r " +
        "WHERE gen=:1 AND r.ID_Regizor=f.Regizori_ID_Regizor",
        (genre,)
    )
    self.topWidget = QWidget()
    self.scrollAreaByGenre[genre].setWidget(self.topWidget)
    self.containerByGenre[genre] = self.topWidget
    self.containerByGenre[genre].setGeometry(QRect(0, 0, 100, 30))
    self.containerByGenre[genre].setObjectName(f"topWidget{genre}")
    topVLayout = QVBoxLayout()
    topVLayout.addLayout(self.makeRow(True, True, 'title', [100, -1, 100, 0], "ID_Film", "Titlu", "An", "Nume Regizor"))
    for text in cursor:
        topVLayout.addLayout(self.makeRow(True, True, 'movie', [100, -1, 100, 0], *text))
    topVLayout.addStretch(1)
    self.containerByGenre[genre].setLayout(topVLayout)
```


Seara de film
HOME PAGE
FAVORITES
NEW MOVIES
MOVIES
ACTORS
MOVIE DIRECTORS
SEARA DE FILM
Action
Adventure
Animated
Comedy
Drama
FANTASY
Horror
Musical
Mystery
Romance
SuperHero
Thriller


Seara de film
HOME PAGE
FAVORITES
NEW MOVIES
MOVIES
ACTORS
MOVIE DIRECTORS
SEARA DE FILM
ACTION MOVIES
ID_Film Titlu An Nume Regizor
3 Gladiator 2000 Scott Ridley Edit Delete
6 Inglorious Basterds 2009 Tarantino Quentin Edit Delete
9 Kill Bill: Volume One 2003 Tarantino Quentin Edit Delete
10 Casino Royale 2006 Campbell Martin Edit Delete
BACK

Exemplu pentru afișarea actorilor din baza de date:

```
def refresh_actors(self):
    self.topWidgetActor.deleteLater()
    self.topWidgetActor = QWidget()
    self.ContainerScroll_Actor.setWidget(self.topWidgetActor)
    self.topWidgetActor.setGeometry(QRect(0, 0, 100, 30))
    self.topWidgetActor.setObjectName("topWidgetActor")
    topVLayout = QVBoxLayout()

    cursor.execute(
        "SELECT a.id_actor, (a.nume||' '||a.prenume), NVL(TO_CHAR(a.data_nasterii), 'Indisponibil'), COUNT(f.id_film) " +
        "FROM Filme f " +
        "RIGHT JOIN Actori a ON f.ACTORI_ID_Actor=a.ID_Actor " +
        "GROUP BY a.id_actor, (a.nume||' '||a.prenume), a.data_nasterii " +
        "ORDER BY id_actor"
    )
    topVLayout.addLayout(self.makeRow(True, True, 'title', [100, -1, 300, 100], "ID Actor", "Nume", "Data nasterii", "Numar Filme"))
    for text in cursor:
        topVLayout.addLayout(self.makeRow(True, True, 'actor', [100, -1, 300, 100], *text))
    topVLayout.addStretch(1)
    self.topWidgetActor.setLayout(topVLayout)
```

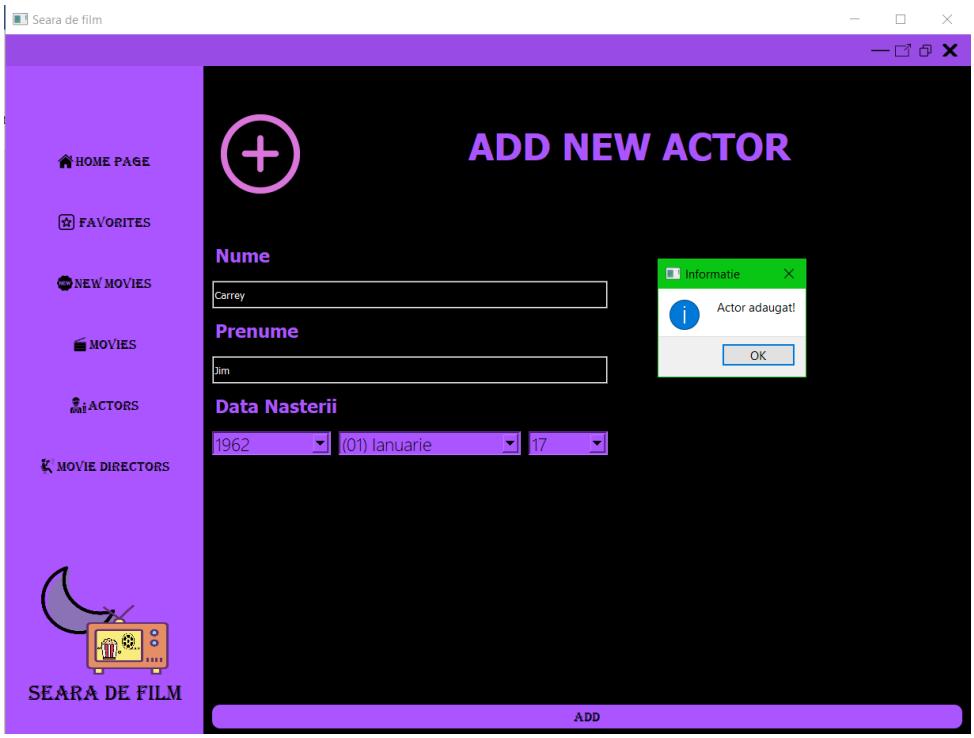
FUNCȚIA INSERT:

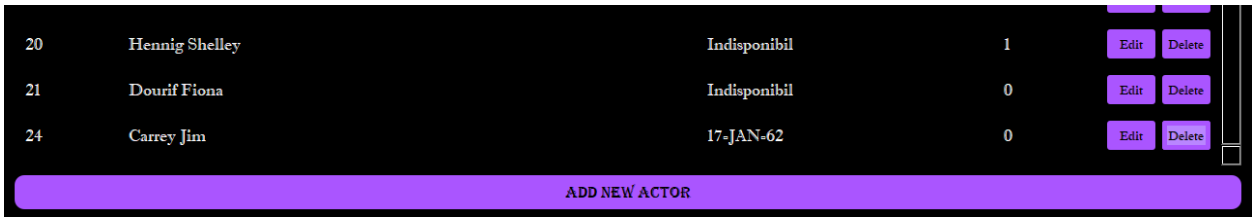
Face posibilă adăugarea unei noi înregistrări.

Exemplu pentru adăugarea unui nou actor:

```
def add_actor(self):
    nume_actor = self.LE_ANA_Nume.text()
    # NULL Test
    if nume_actor == "":
        self.openDialogBox('Numele este invalid! Trebuie sa fie diferit de NULL!', 'warning')
        return
    for char in nume_actor:
        if char.isdigit():
            self.openDialogBox('Numele este invalid! Nu ar trebui sa contina cifre!', 'warning')
            return
        if not char.isalpha() and not char.isdigit():
            self.openDialogBox('Numele este invalid! Nu ar trebui sa contina simboluri!', 'warning')

    prenume_actor = self.LE_ANA_Prenume.text()
    if prenume_actor != "":
        for char in prenume_actor:
            if char.isdigit():
                self.openDialogBox('Prenumele este invalid! Nu ar trebui sa contina cifre!', 'warning')
                return
            if not char.isalpha() and not char.isdigit():
                self.openDialogBox('Prenumele este invalid! Nu ar trebui sa contina simboluri!', 'warning')

    an = self.comboBoxA_an.currentText()
    luna = self.comboBoxA_luna.currentText()
    zi = self.comboBoxA_zi.currentText()
    data_nasterii_actor = an + '-' + luna[1:3] + '-' + zi
```

```
try:
    datetime.datetime(int(an), int(luna[1:3]), int(zi))
except ValueError:
    self.openDialogBox('Data de nastere nu este valida! Verifica ziua!', 'warning')
    return

cursor.execute(
    "INSERT INTO actori(nume, prenume, data_nasterii) VALUES(:1, :2, TO_DATE(:3, 'YYYY-MM-DD'))",
    (nume_actor, prenume_actor, data_nasterii_actor)
)
connection.commit()

self.openDialogBox('Actor adaugat!')
self.LE_ANA_Nume.setText("")
self.LE_ANA_Prenume.setText("")
```

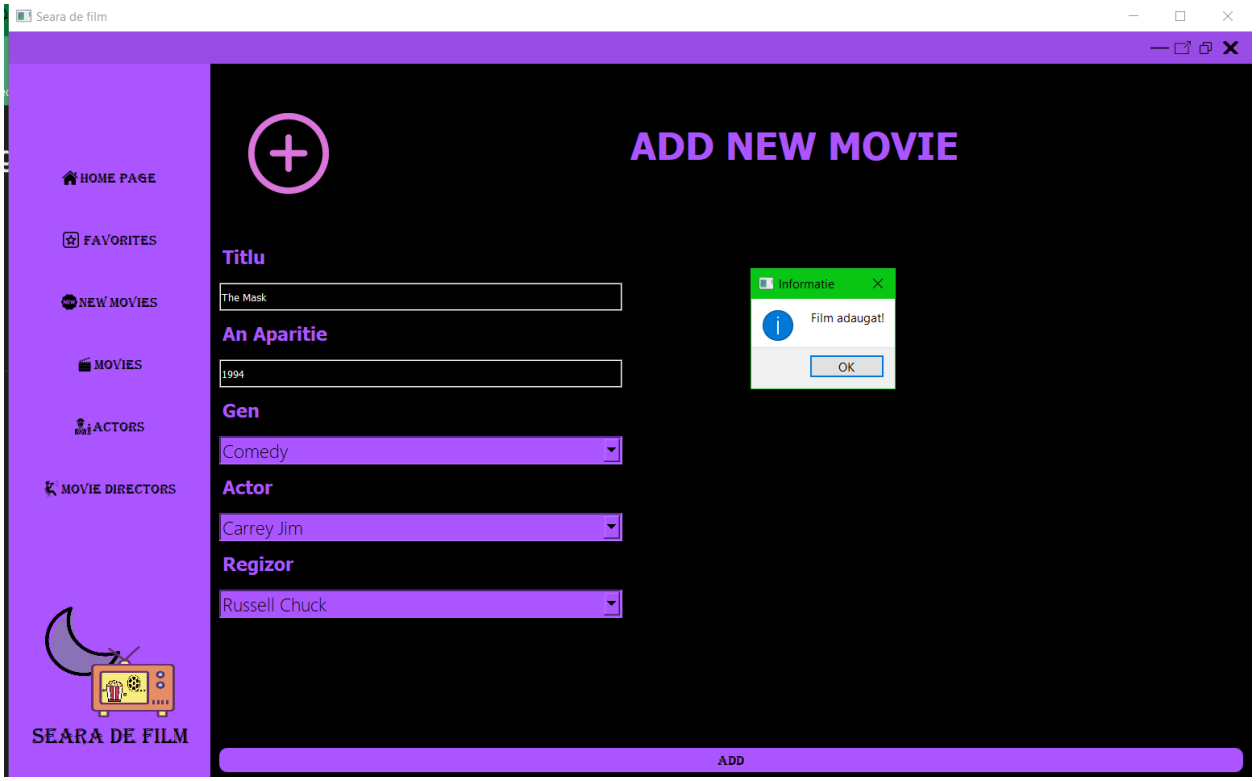




Exemplu pentru adăugarea unui nou film:

```
def add_film(self):
    titlu_film = self.LE_ANM_Titlu.text()
    # NULL Test
    if titlu_film == "":
        self.openDialogBox('Titlul este invalid! Trebuie sa fie diferit de NULL!', 'warning')
        return

    an_ap_film = self.LE_ANM_AnAparitie.text()
    if not an_ap_film.isdigit():
        self.openDialogBox('Introdu un an valid!')
        return
    an_ap_film = int(an_ap_film)

    gen_film = self.comboBoxAdd_Gen.currentText()

    nume_actor = self.comboBoxAdd_Actor.currentText()
    cursor.execute("select id_actor, nume, prenume from actori")
    for elem in cursor:
        if str(elem[1]) + ' ' + str(elem[2]) == nume_actor:
            id_actor_film = int(elem[0])
            break

    nume_regizor = self.comboBoxAdd_Regizor.currentText()
    cursor.execute("select id_regizor, nume, prenume from regizori")
    for elem in cursor:
        if str(elem[1]) + ' ' + str(elem[2]) == nume_regizor:
            id_regizor_film = int(elem[0])
            break
```

```
cursor.execute(
    "INSERT INTO filme(titlu, an_aparitie, gen, actori_id_actor, regizori_id_regizor, isTop) " +
    "VALUES(:1, :2, :3, :4, :5, :6)",
    (titlu_film, an_ap_film, gen_film, id_actor_film, id_regizor_film, 0)
)
connection.commit()

self.openDialogBox('Film adaugat!')
self.LE_ANM_Titlu.setText("")
self.LE_ANM_AnAparitie.setText("")
```


Seara de film
HOME PAGE
FAVORITES
NEW MOVIES
MOVIES
ACTORS
MOVIE DIRECTORS
SEARA DE FILM
ADD NEW MOVIE
Titlu
The Mask
An Aparitie
1994
Gen
Comedy
Actor
Carrey Jim
Regizor
Russell Chuck
ADD
Informatie
Film adaugat!
OK

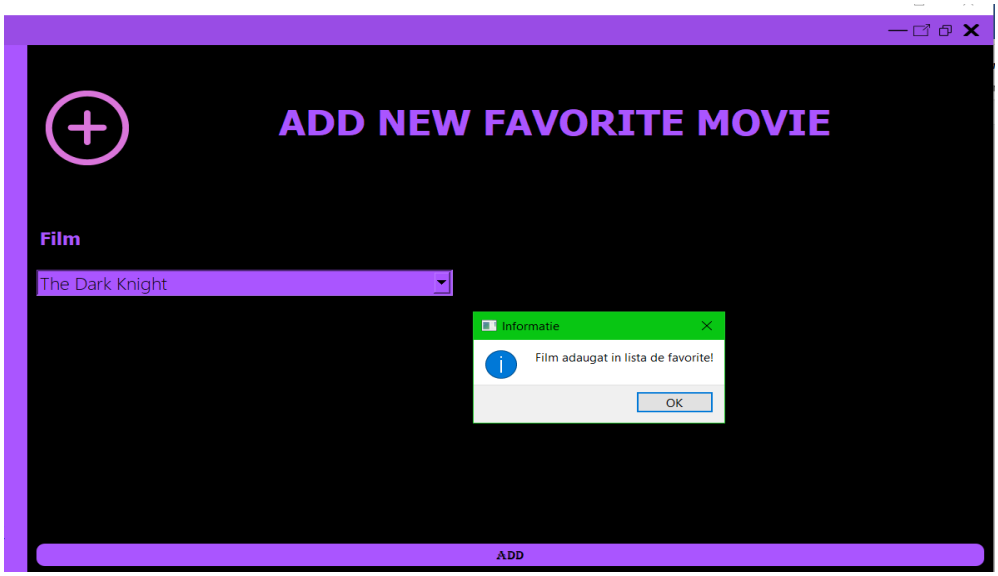

COMEDY MOVIES
ID_Film Titlu An Nume Regizor
23 Naked 2017 Tiddes Michael Edit Delete
30 The Mask 1994 Russell Chuck Edit Delete


24 Carrey Jim 17-JAN-62 1 Edit Delete


25 Russell Chuck 09-MAY-58 1 Edit Delete

Exemplu inserare în lista de favorite:

```
def add_favorite(self):
    nume_film = self.comboBoxAdd_Fav.currentText()

    cursor.execute("select id_film, titlu from filme")
    for elem in cursor:
        if str(elem[1]) == nume_film:
            cursor2.execute("select filme_id_film from favorite")
            for elem2 in cursor2:
                if int(elem2[0]) == int(elem[0]):
                    self.openDialogBox('Film deja existent in lista ta de favorite!', 'warning')
                    return

    cursor.execute("select id_film, titlu from filme")
    for elem in cursor:
        if str(elem[1]) == nume_film:
            id_favMovie = int(elem[0])
            break

    # Adauga filmul in lista de favorite
    cursor.execute(f"INSERT INTO favorite SELECT f.id_film FROM filme f WHERE f.id_film = {id_favMovie}")
    connection.commit()

    self.openDialogBox('Film adaugat in lista de favorite!')
```



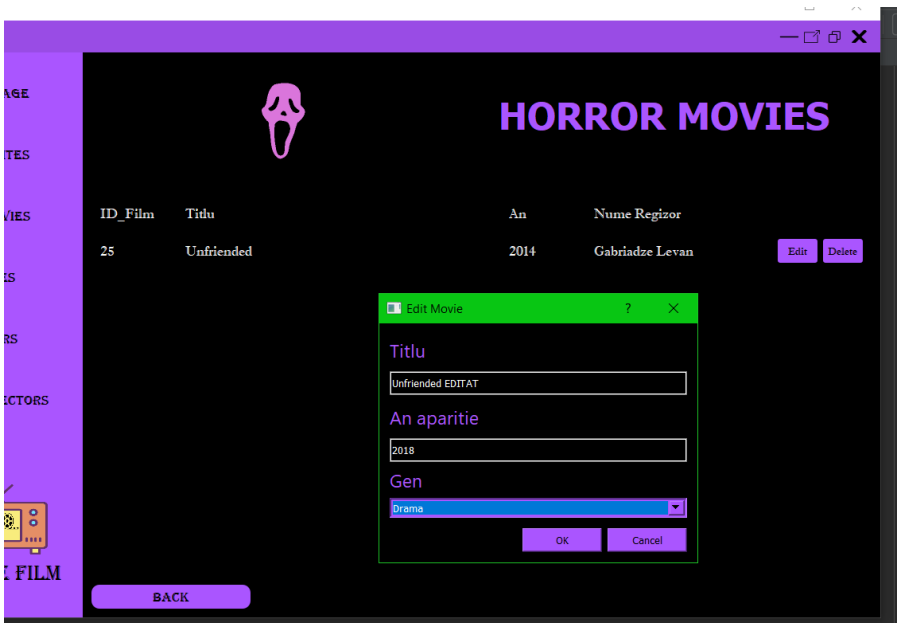
FUNCȚIA UPDATE:

Modifica datele existente cu alte date valide.

Exemplu pentru editarea datelor unui film:

```
def editEntry(self, rowType: str, *args):
    print(rowType)
    print(args)
    self.openDialogInput(rowType, args)
    if not self.confirmUpdate:
        return
    if rowType == 'movie':
        cursor.execute(
            "UPDATE filme SET titlu = :1, an_aparitie = :2, gen = :3 WHERE id_film = :4",
            (self.movieData['Titlu'], self.movieData['AnAparitie'], self.movieData['Gen'], args[0])
        )
        connection.commit()
        self.refreshAllGenres()
    elif rowType == 'actor':
        cursor.execute(
            "UPDATE actori SET nume = :1, prenume = :2, data_nasterii = TO_DATE(:3, 'YYYY-MM-DD') WHERE id_actor = :4",
            (self.actorData['Nume'], self.actorData['Prenume'], self.actorData['DataNasterii'], args[0])
        )
        connection.commit()
        self.refresh_actors()
    elif rowType == 'moviedirector':
        cursor.execute(
            "UPDATE regizori SET nume = :1, prenume = :2, data_nasterii = TO_DATE(:3, 'YYYY-MM-DD') WHERE id_regizor = :4",
            (self.regizorData['Nume'], self.regizorData['Prenume'], int(self.regizorData['DataNasterii']), args[0])
        )
        connection.commit()
```

```
    connection.commit()
    self.refresh_movieDirectors()
self.confirmUpdate = False
print('am trecut de confirm')
```




HORROR MOVIES
ID_Film
Titlu
An
Nume Regizor
BACK


DRAMA MOVIES
ID_Film
Titlu
An
Nume Regizor
18
Edward Scissorhands
1991
Burton Tim
Edit
Delete
25
Unfriended EDITAT
2018
Gabriadze Levan
Edit
Delete
BACK

FUNCȚIA DELETE:

Șterge o înregistrare din tabel.

Exemplu pentru ștergerea unui film:

```
def deleteEntry(self, rowType: str, *args):
    print(rowType)
    print(args)
    if rowType == 'movie':
        cursor.execute("select gen from filme where id_film = :1", (args[0],))
        for e in cursor:
            gen = e[0]
        cursor.execute("DELETE FROM filme WHERE :1 = id_film", (args[0],))
        connection.commit()
        self.refresh_genrePage(gen)
    elif rowType == 'actor':
        if args[3] > 0:
            self.openDialogBox('Nu se poate sterge actorul! El joaca intr-ul film din baza de date!', 'warning')
            return
        cursor.execute("DELETE FROM actori WHERE :1 = id_actor", (args[0],))
        connection.commit()
        self.refresh_actors()
    elif rowType == 'moviedirector':
        if args[3] > 0:
            self.openDialogBox('Nu se poate sterge regizorul! El a facut un film care se afla in baza de date!', 'warning')
            return
        cursor.execute("DELETE FROM regizori WHERE :1 = id_regizor", (args[0],))
        connection.commit()
        self.refresh_movieDirectors()
    elif rowType == 'fav':
        cursor.execute("DELETE FROM favorite WHERE :1 = filme_id_film", (args[0],))
        connection.commit()
        self.refresh_fav()
```