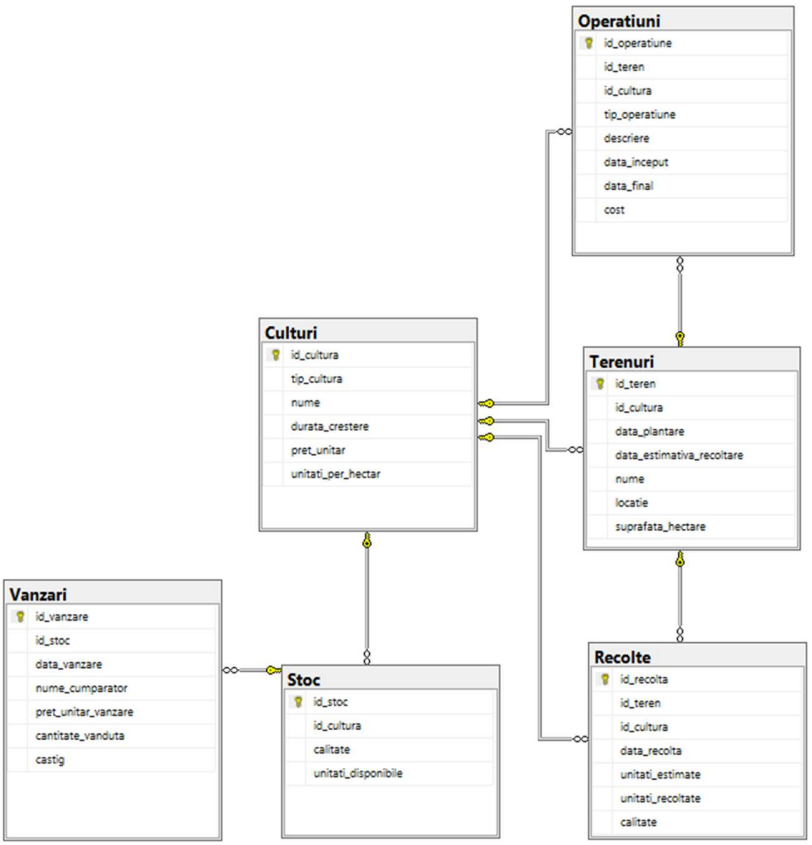
Proiect Baze de Date 2

- Studenta: ANA Elena-Diana
  Grupa: 342C3

- Descrierea temei

Proiectul meu reprezinta baza unui sistem de gestiune pentru o ferma agricola. Scopul proiectului este de a ajuta user-ul sa gestioneze activitatile aferente procesului agricol, precum: evidenta culturilor, a terenurilor, inregistrarea vanzarilor, monitorizarea operatiunilor si a stocurilor disponibile.

- Descrierea bazei de date
  - Diagrama bazei de date:



- Structura tabelelor:

Tabelul *Culturi* contine informatii despre toate plantele produse in ferma, precum: durata de crestere, pretul de baza per unitate si numarul de unitati care cresc intr-un hectar de pamant, in conditii normale. Acestea reprezinta datele de referinta pentru rapoartele de productie si vanzari.

Tabelul *Terenuri* tine evidenta tuturor terenurilor fermei, precum si ce culturi sunt in proces de crestere pe acestea.

Tabelul *Operatiuni* contine date despre operatiunile agricole efectuate pe terenuri, precum: ararea, plantarea, recoltarea etc.

Tabelul *Recolte* inregistreaza date despre fiecare recolta, dupa o operatiune de recoltare, precum: numarul de unitati estimate ale recoltei, unitatile reale/recoltate si calitatea recoltei.

Tabelul *Stoc* tine evidenta unitatilor disponibile pentru vanzare, pentru fiecare cultura si calitatile sale aferente.

Tabelul *Vanzari* contine informatii despre vanzarile efectuate precum: pretul de vanzare, care depinde de pretul de baza al culturii si calitatea sa, cantitatea vanduta, care trebuie sa existe in stoc si castigul obtinut in urma vanzarii.

- Descrierea constrangerilor de integritate:

FOREIGN KEYS – asigura relatiile dintre tabele

CHECK – adauga constrangeri pe valorile coloanelor (de exemplu: intervalul calitatii [0.1, 1], valoarea pozitiva pentru unitatile recoltate, multimea de valori acceptate pentru tipul culturii {'leguma', 'fruct', 'cereala', 'iarba_aromatica', 'floare'})

UNIQUE – asigura claritatea datelor prin conditia de unicitate a anumitor denumiri (denumiri de terenuri, de plante/culturi)

NOT NULL – asigura ca anumite campuri importante (precum: data recoltarii, tipul operatiunii) nu sunt lasate necompletate

TRIGGERE – asigura actualizari automate a unor tabele

- *TRG-plantare-teren*: actualizeaza tabela Terenuri, setand data_plantare, data_estimativa_recoltare si id_cultura dupa inserarea unei operatiuni de tip plantare in tabela Operatiuni
- *TRG-pret-vanzare*: calculeaza automat pretul unitar al unei vanzari pe baza calitatii stocului din care s-a facut vanzarea si pretul de baza al culturii

- *TRG-recoltare-stoc*: actualizeaza tabela Stoc prin introducerea/modificarea unei intrari in functie de cultura recoltata si calitatea acesteia
- *TRG-recoltare-teren*: elimina data_plantare, data_estimativa_recoltare si id_cultura de pe teren dupa finalizarea unei operatiuni de recoltare
- *TRG-vanzare-stoc*: scade cantitatea vanduta din stocul disponibil atunci cand se introduce o vanzare in tabela Vanzari

➢ Descrierea procedurilor:

1. *RaportPerformanteTerenuri*:
    - raport pentru anul 2023 care arata performanta terenurilor dpdv cantitativ si calitativ in raport cu numarul operatiunilor de intretinere efectuate
    - userul poate lua decizii de sporire a productivitatii in functie de cat de mult influenteaza operatiile de intretinere recolta obtinuta
2. *RaportPerformanteCulturi*:
    - raport pentru anul 2023 care arata, pentru fiecare cultura in parte, pe ce teren a crescut cel mai bine dpdv cantitativ si care a fost pretul de productie pe acel teren
    - userul poate lua decizii de plantare a culturilor viitoare in functie de care teren este mai potrivit pentru fiecare cultura
3. *RaportVanzariCalitate*:
    - raport pentru anul 2024 care arata, pentru fiecare cultura in parte, pentru fiecare categorie de calitate ([0.1, 0.5] mediocra [0.5, 0.7] buna [0.7, 0.9] foarte buna [0.9, 1] excelenta) care a fost castigul vanzarilor, cat % din vanzarile totale ale culturii respective a insemnat fiecare categorie si ce cumparatori prefera fiecare categorie (au cumparat cel mai mult dpdv cantitativ din acea cagetorie)
    - userul poate lua decizii de optimizare a calitatii produse pentru fiecare cultura in parte in functie de cererea pietei

Descrierea aplicatiei

Aplicatia reprezinta un script Python de creare si afisare de rapoarte pentru baza de date, care se conecteaza la aceasta, executa proceduri stocate si afiseaza datele obtinute in format grafic.

➢ Structura claselor:

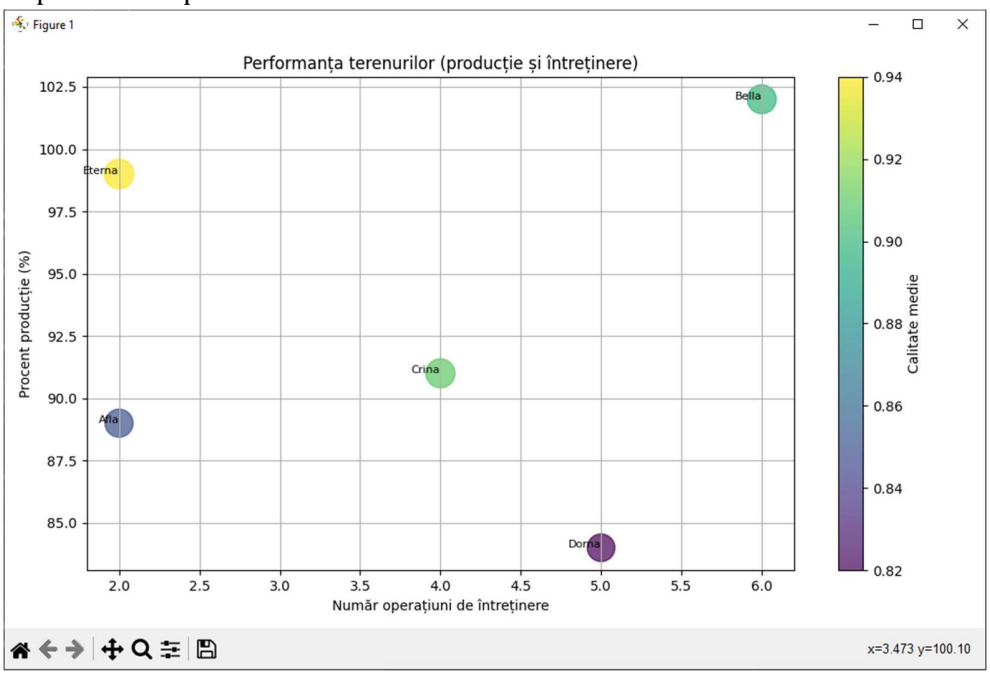
Clasa principala este MSSQLConnection care contine:

- *Atribute*: host, database, username, password, db, cursor
- *Metode*: openConnection(), closeConnection(), getRaportPerformanteTerenuri(), getRaportPerformanteCulturi(), getRaportVanzariCalitate()

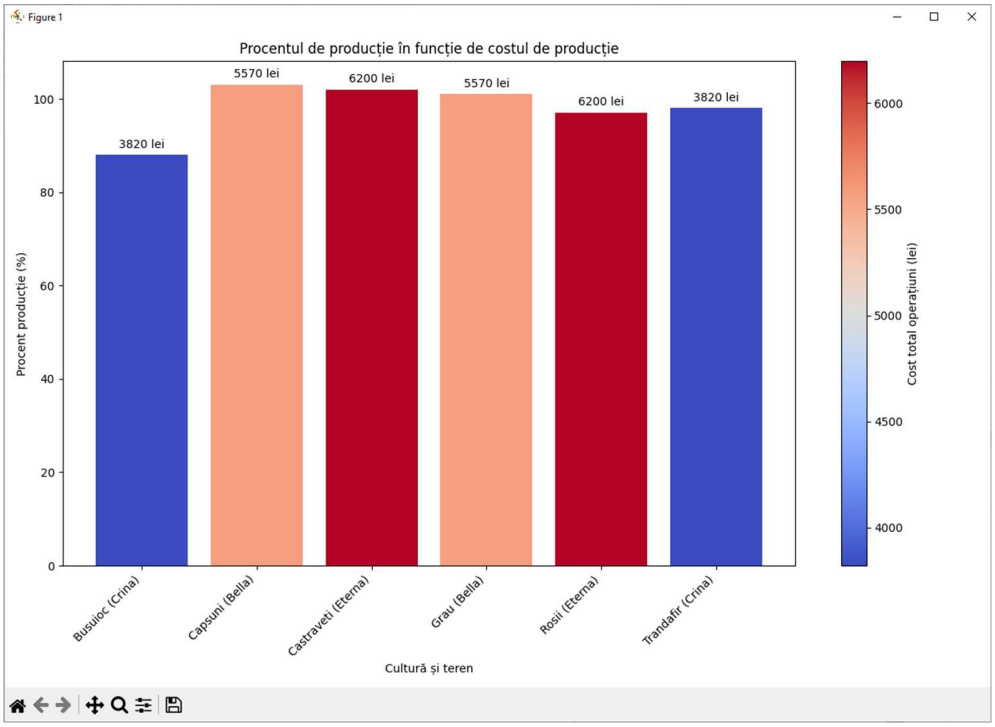
- Workflow-ul aplicatiei
  1. *Conectarea la baza de date*
     - Clasa MSSQLConnection gestioneaza conectarea si deconectarea de la baza de date folosind pyodbc.
     - Parametrii conexiunii includ host, database, username, password.
     - Metodele openConnection() si closeConnection() deschid si inchid conexiunea.
  2. *Executarea rapoartelor*
     - Fiecare metoda din clasa MSSQLConnection (getRaportPerformanteTerenuri(), getRaportPerformanteCulturi(), getRaportVanzariCalitate()) apeleaza cate o procedura SQL stocata (EXEC Raport...) pentru a prelua datele din baza de date.
  3. *Afisarea datelor*
     - Functiile plotRaportPerformanteTerenuri(), plotRaportPerformanteCulturi(), plotRaportVanzariCalitate() transforma datele obtinute in grafice folosind biblioteca matplotlib.
     - Graficele prezinta rezultate sub forma de diagrame verticale sau scatter plots pentru a analiza performanta terenurilor, eficienta culturilor si distributia vanzarilor pe categorii de calitate.

- Folosirea aplicatiei
  1. Pornire container Docker SQL Server
  2. Conectare SSMS
  3. Rulare scripturi de creare tabele, triggere si proceduri in SSMS – script.sql, raport-complex-4.sql, raport-complex-6.sql, raport-complex-7.sql
  4. Rulare script Python – main.py

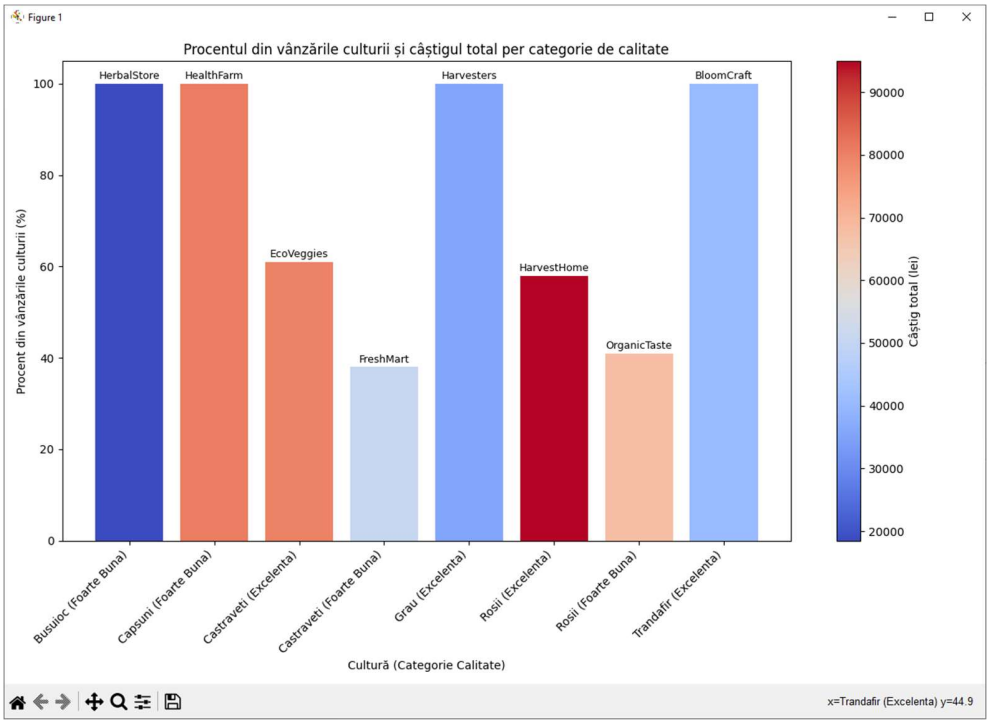
- Capturi de ecran pentru rapoarte
  - Raport de complexitate 4:



  - Raport de complexitate 6:



- Raport de complexitate 7:



Concluzii

- Proiectul implementeaza o baza de date pentru o ferma agricola, care permite urmarirea productiei, a operatiunilor si a vanzarilor.
- Rapoartele generate permit analizarea performantei culturilor si terenurilor, oferind informatii relevante pentru luarea deciziilor de optimizare.
- Implementarea scriptului Python asigura automatizarea preluarii datelor si afisarea vizuala a rapoartelor.
- Solutia dezvoltata demonstreaza un flux logic coerent: de la inserarea datelor pana la obtinerea rapoartelor grafice, fiind o baza solida pentru eventuale extensii, cum ar fi adaugarea unei interfete grafice si a mai multor proceduri SQL.

Bibliografie

- Documentatia oficiala Microsoft SQL Server: https://learn.microsoft.com/en-us/sql
- Documentatia pyodbc: https://github.com/mkleehammer/pyodbc
- Documentatia matplotlib: https://matplotlib.org/stable/index.html