

Proiect Baze de date

Nuță Leonard-Florian
Facultatea de Automatică și Calculatoare
Grupa 333AA

Descrierea cerinței

Aplicație pentru evidența unei companii de concerte

Ipoteza : Compania ține evidența tuturor concertelor ce se desfășoară în diferite **localități**, pe mai multe zile. Fiecare categorie de persoană beneficiază de o reducere: **studenții** reducere 50%, **vârstnicii** reducere 25%. La fiecare **concert** cântă un număr de **artiști** astfel încât timpul rezervat să fie respectat.

Reguli:

La un **concert** cântă mai mulți **artiști**.

Un **artist** cântă la mai multe **concerte**.

Avem relație de N:N; realizăm tabelul de legătură **ArtistConcert**.

Un **bilet** poate fi achiziționat de un singur **cumpărător**.

Un **cumpărător** poate lua mai multe **bilete**. (Este vorba de cazul în care persoana cumpără bilete la diferite concerte)

Avem relație de 1:N; cumpărător – unul; bilete – mai mulți (cheia de la **Cumpărător** se trece la **Bilet**)

Un **bilet** se cumpără pentru un **concert**.

Un **concert** vinde mai multe **bilete**.

Avem relație de 1:N; concert – unul; bilete – mai mulți; (cheia de la **Concert** se trece la **Bilet**)

Un **concert** se poate desfășoară în diferite **localități**.

Într-o **localitate** pot avea loc mai multe **concerte**.

Avem relație de N:N; Realizăm tabelul de legătură **ConcertLocalitate**.

Avem **tabelele** :

- Cumparator(CumparatorID, Nume, Prenume, CNP, CategoriePersoana(student/nu - D/N))
- Bilet(BiletID, CumparatorID, ConcertID, Data, Pret, NumarLoc)
- Concert(ConcertID, NumeConcert, DataInceput, DataSfarsit, NumarLocuri, Data)
- Artist(ArtistID, Nume, Prenume, TipMuzica)
- ArtistConcert(ArtistID, ConcertID, NumarPiese)
- Localitate(LocalitateID, Denumire, Judet)
- ConcertLocalitate(ConcertID, LocalitateID)

Entități:

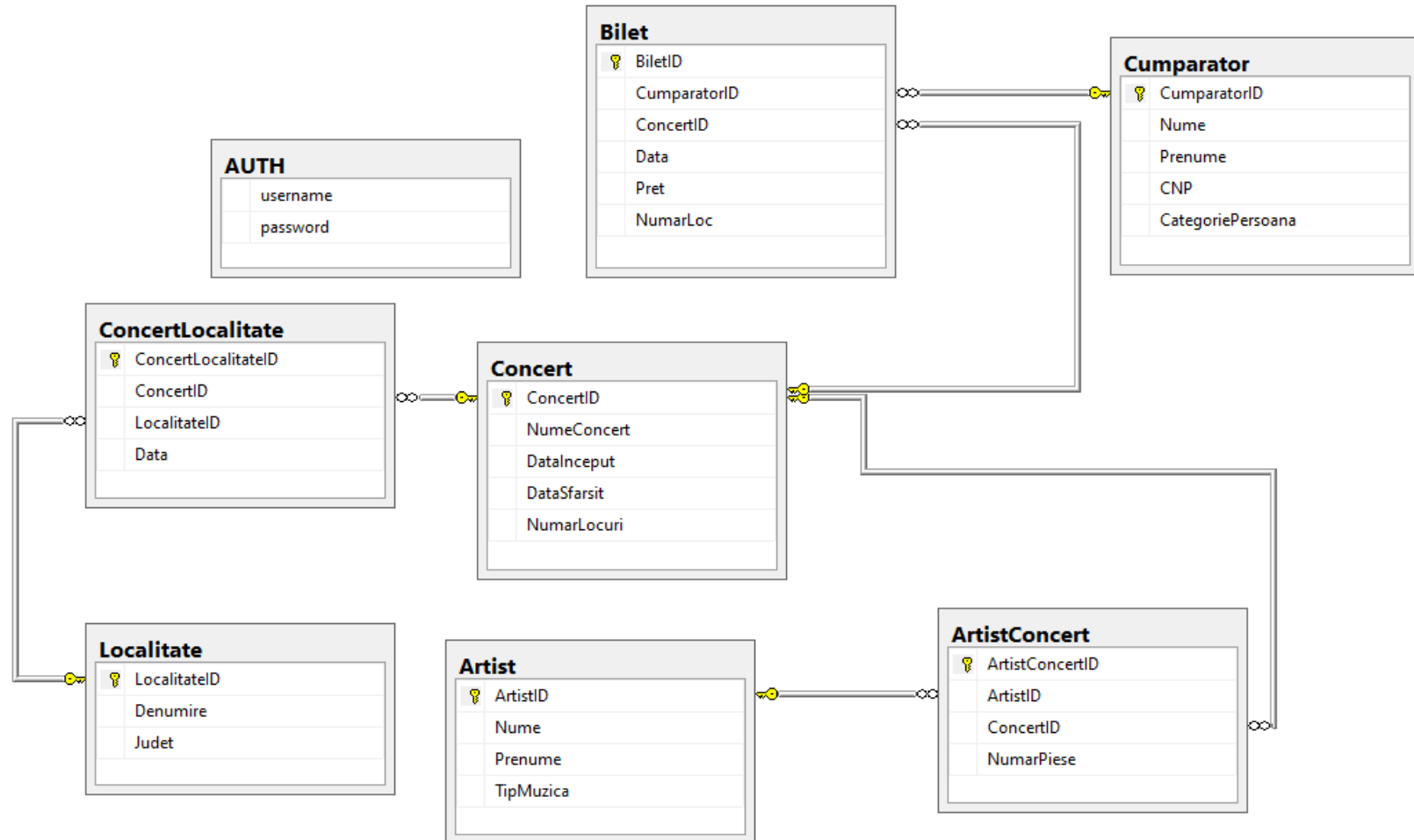
- Cumpărător
- Bilet
- Concert
- Artist
- Localitate

Constrângeri:

- CategoriePersoana poate lua doar valorile de D sau N, altfel întâmpinăm eroare.
- CNP este unic.

Baza de date îndeplinește condițiile de Forma Normală de Ordinul 3 (care implică FN1 și FN2) întrucât toate câmpurile conțin date atomice, nu există attribute compuse, nu există câmpuri ce rezultă din derivarea mai multora și toate tabelele nu au date duplicate.

Diagrama relațională



Partea de interfață a aplicației bazei de date

Aplicația bazei de date a fost realizată în Visual Studio 2022, limbajul de programare utilizat fiind C#.

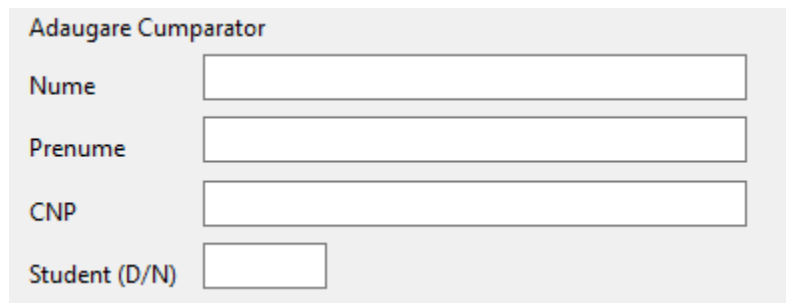
Pentru realizarea conexiunii la baza de date, am indicat în file-ul settings.settings connection string-ul care este identic cu cel al bazei noastre de date :

```
connectionString="Data Source=DESKTOP-EV1SM1J\SQLEXPRESS;Initial  
Catalog=Proiect_Nuta_Leonard_Florian_333AA;Integrated Security=True"
```

Operații asupra bazei de date

Am realizat operațiile de **insert, update, delete** pentru tabelele **Cumpărător** și **Artist**.

Pentru **Cumpărător**:



The screenshot shows a web form titled "Adaugare Cumparator". It has four input fields with labels to their left: "Nume", "Prenume", "CNP", and "Student (D/N)". Each label and its corresponding input field are enclosed in a light gray rounded rectangle. The "Student (D/N)" field is smaller than the others.

Ce se tastează în caseta pentru Nume va fi salvat în parametrul @Nume.

Ce se tastează în caseta pentru Prenume va fi salvat în parametrul @Prenume.

Ce se tastează în caseta pentru CNP va fi salvat în parametrul @CNP.

Ce se tastează în caseta pentru Student va fi salvat în parametrul @CategoriePersoana.

Query-ul din cod este:

```
INSERT INTO Cumparator (Nume, Prenume, CNP, CategoriePersoana) VALUES (@Nume,  
@Prenume, @CNP, @CategoriePersoana)
```

și are sintaxa specifică SQL, doar că parametrii vor fi introduși de la tastatură, în interfață, în loc să avem ceva de genul:

```
INSERT INTO Cumparator (Nume, Prenume, CNP, CategoriePersoana) VALUES ("Nuță", "Leonard",  
"5021223xxxxxx", "D")
```

Parametrii sunt variabili fiindcă acesta este și rolul unei interfețe pentru adăugarea unor rânduri în tabel.

Eliminare Cumparator

CNP

DELETE FROM Cumparator WHERE CNP = @CNP

Editare Cumparator

Nume

Prenume

CNP

CNP Vechi

Student (D/N)

UPDATE Cumparator SET Nume = @Nume, Prenume = @Prenume, CNP = @CNP,
CategoriePersoana = @CategoriePersoana WHERE CNP = @CNP2

Aici, ce se tastează în caseta CNP Vechi va fi salvat în parametrul @CNP2. Update-ul rândului se va face după un CNP, CNP pe care urmează să-l înlocuim cu același sau unul nou, împreună cu modificarea celorlalte date: Nume, Prenume, Student(D/N)

Pentru **Artist**:

Adaugare Artist

Nume

Prenume

Tip Muzica

INSERT INTO Artist (Nume, Prenume, TipMuzica) VALUES (@Nume, @Prenume, @TipMuzica)

Eliminare Artist

Nume

Prenume

DELETE FROM Artist WHERE (Nume = @Nume AND Prenume IS NULL) OR (Nume = @Nume AND Prenume = @Prenume)

Eliminarea se va face după Nume și Prenume, întrucât în cerința temei se menționează faptul că nu avem permisiunea de a folosi câmpuri irelevante precum ID.

Editare Artist

Nume	<input type="text"/>	Nume Vechi	<input type="text"/>
Prenume	<input type="text"/>	Prenume Vechi	<input type="text"/>
Tip Muzica	<input type="text"/>		

UPDATE Artist SET Nume = @Nume, Prenume = @Prenume, TipMuzica = @TipMuzica WHERE (Nume = @Nume2 AND Prenume IS NULL) OR (Nume = @Nume2 AND Prenume = @Prenume2)

Ce se tastează în caseta Nume Vechi va fi salvat în parametrul Nume2.

Ce se tastează în caseta Prenume Vechi va fi salvat în parametrul Prenume2.

Mai exact, căutam un artist după Nume și Prenume, iar apoi îi înlocuim toate datele cu cele din casetele Nume, Prenume, TipMuzica (@Nume, @Prenume, @TipMuzica).

Interogările simple:

Listăți toate concertele cu artiștii corespunzători:

Găsiți toți cumpărătorii care au cumpărat bilete și concertele la care au participat:

Preluăți informații despre concert împreună cu locațiile în care au avut loc:

Listăți toate concertele cu numărul de melodii pe care le va interpreta fiecare artist:

Afișați toate detaliile biletului, inclusiv informații despre cumpărător și detalii despre concert:

Preluăți toate concertele dintr-un anumit județ împreună cu artiștii care interpretează:

Județ

1. Listați toate concertele cu artiștii corespunzători:

Această interogare preia o listă de concerte împreună cu artiștii corespunzători care vor cânta în fiecare concert.

```
SELECT Concert.ConcertID, Concert.NumeConcert, Artist.Nume, Artist.Prenume
FROM Concert
INNER JOIN ArtistConcert ON Concert.ConcertID = ArtistConcert.ConcertID
INNER JOIN Artist ON ArtistConcert.ArtistID = Artist.ArtistID;
```

2. Găsiți toți cumpărătorii care au cumpărat bilete și concertele la care au participat:

Această interogare listează cumpărătorii împreună cu concertele la care au participat și data la care au achiziționat biletele.

```
SELECT Cumparator.Nume, Cumparator.Prenume, Concert.NumeConcert, Bilet.Data
FROM Cumparator
INNER JOIN Bilet ON Cumparator.CumparatorID = Bilet.CumparatorID
INNER JOIN Concert ON Bilet.ConcertID = Concert.ConcertID;
```

3. Preluați informații despre concert împreună cu locațiile în care au avut loc:

Această interogare combină informațiile despre concert cu locațiile în care au loc concertele.

```
SELECT Concert.NumeConcert, Concert.DataInceput, Concert.DataSfarsit, Localitate.Denumire,
Localitate.Judet
FROM Concert
INNER JOIN ConcertLocalitate ON Concert.ConcertID = ConcertLocalitate.ConcertID
INNER JOIN Localitate ON ConcertLocalitate.LocalitateID = Localitate.LocalitateID;
```

4. Listați toate concertele cu numărul de melodii pe care le va interpreta fiecare artist:

Această interogare oferă informații despre concerte și numărul de melodii pe care le va interpreta fiecare artist.

```
SELECT Concert.ConcertID, Concert.NumeConcert, Artist.Nume, Artist.Prenume,
ArtistConcert.NumarPiese
FROM Concert
INNER JOIN ArtistConcert ON Concert.ConcertID = ArtistConcert.ConcertID
INNER JOIN Artist ON ArtistConcert.ArtistID = Artist.ArtistID;
```


5. Afișați toate detaliile biletului, inclusiv informații despre cumpărător și detalii despre concert:

Această interogare afișează informații detaliate despre fiecare bilet, inclusiv detalii despre cumpărător și informații despre concert.

```
SELECT Bilet.BiletID, Bilet.Data, Bilet.Pret, Bilet.NumarLoc, Cumparator.Nume,
Cumparator.Prenume, Concert.NumeConcert
FROM Bilet
INNER JOIN Cumparator ON Bilet.CumparatorID = Cumparator.CumparatorID
INNER JOIN Concert ON Bilet.ConcertID = Concert.ConcertID;
```

6. Preluati toate concertele dintr-un anumit județ împreună cu artiștii care interpretează:

Această interogare preia concertele susținute într-un anumit județ și artiștii corespunzători.

```
SELECT Concert.NumeConcert, Artist.Nume, Artist.Prenume, Localitate.Judet
FROM Concert
INNER JOIN ConcertLocalitate ON Concert.ConcertID = ConcertLocalitate.ConcertID
INNER JOIN Localitate ON ConcertLocalitate.LocalitateID = Localitate.LocalitateID
INNER JOIN ArtistConcert ON Concert.ConcertID = ArtistConcert.ConcertID
INNER JOIN Artist ON ArtistConcert.ArtistID = Artist.ArtistID
WHERE Localitate.Judet = @Judet;
```

Unde @Judet este parametrul introdus în casuța pentru Județ.

Interogări complexe:

Găsiți concertul cu cel mai mare număr de participanți:	START
Listați toate concertele la care prețul mediu al biletului este mai mare decât media generală:	START
Afișați toți cumpărătorii care au participat la aceleași concerte ca un anumit cumpărător:	START
CNP	<input type="text"/>
Preluati concerte în care numărul de melodii interpretate de artiști este mai mare decât media generală:	START

1. Găsiți concertul cu cel mai mare număr de participanți:

Această interogare identifică concertul cu cel mai mare număr de participanți.

```
SELECT Concert.NumeConcert, COUNT(Bilet.BiletID) AS NumarParticipanti
FROM Concert
LEFT JOIN Bilet ON Concert.ConcertID = Bilet.ConcertID
GROUP BY Concert.NumeConcert
HAVING COUNT(Bilet.BiletID) = (
    SELECT MAX(NumarParticipanti)
    FROM (
        SELECT ConcertID, COUNT(BiletID) AS NumarParticipanti
        FROM Bilet
        GROUP BY ConcertID
    ) AS ParticipantiConcert
);
```

2. Listați toate concertele la care prețul mediu al biletului este mai mare decât media generală:

Această interogare identifică concertele cu un preț mediu al biletului mai mare decât media generală.

```
SELECT Concert.NumeConcert, AVG(Bilet.Pret) AS AvgTicketPrice
FROM Concert
LEFT JOIN Bilet ON Concert.ConcertID = Bilet.ConcertID
GROUP BY Concert.NumeConcert
HAVING AVG(Bilet.Pret) > (SELECT AVG(Pret) FROM Bilet);
```

3. Afișați toți cumpărătorii care au participat la aceleași concerte ca un anumit cumpărător:

Această interogare găsește alți cumpărători care au participat la aceleași concerte ca și un cumpărător specificat după CNP.

```
SELECT DISTINCT AltCumparator.Nume, AltCumparator.Prenume
FROM Bilet AS BiletPrincipal
INNER JOIN Cumparator AS CumparatorPrincipal ON BiletPrincipal.CumparatorID =
CumparatorPrincipal.CumparatorID
INNER JOIN (
    SELECT Bilet.ConcertID, Cumparator.CumparatorID
    FROM Bilet
    INNER JOIN Cumparator ON Bilet.CumparatorID = Cumparator.CumparatorID
    WHERE Cumparator.CNP = @CNP
) AS Subquery ON BiletPrincipal.ConcertID = Subquery.ConcertID
INNER JOIN Bilet AS AltBilet ON BiletPrincipal.ConcertID = AltBilet.ConcertID
INNER JOIN Cumparator AS AltCumparator ON AltBilet.CumparatorID =
AltCumparator.CumparatorID;
```

4. Preluați concerte în care numărul de melodii interpretate de artiști este mai mare decât un anumit prag:

Această interogare preia concertele în care numărul de melodii interpretate de artiști depășește numărul mediu de melodii din toate concertele.

```
SELECT Concert.NumeConcert
FROM Concert
INNER JOIN ArtistConcert ON Concert.ConcertID = ArtistConcert.ConcertID
WHERE ArtistConcert.NumarPiese > (SELECT AVG(NumarPiese) FROM ArtistConcert);
```