

## Proiect Baze de Date(etapa 1)

Gradinaru Cristian 332AC

Descrierea Cerintelor: Avem nevoie de o baza de date pentru un centru de inchierere a bicicletelor. Pentru aceasta m-am hotarat in folosirea a 6 tabele dupa cum urmeaza:

### Tabela Angajati:

ID\_Angajat fiind o cheie primara, unica, cu autoincrementare: cu ajutorul ei vom identifica fiecare angajat din cadrul companiei intr-o maniera eficienta si clara. Alte campuri sunt de tipul Nume si Prenume, CNP si Telefon. Am ales sa nu permit un camp nul pentru CNP, pentru ca in acest fel pot oricand identifica un angajat fara sa existe confuzii de nume.

|   | Column Name | Data Type   | Allow Nulls                         |
|---|-------------|-------------|-------------------------------------|
| 🔑 | ID_Angajat  | int         | <input type="checkbox"/>            |
|   | Nume        | varchar(50) | <input checked="" type="checkbox"/> |
|   | Prenume     | varchar(50) | <input checked="" type="checkbox"/> |
|   | CNP         | varchar(50) | <input type="checkbox"/>            |
|   | Telefon     | varchar(50) | <input checked="" type="checkbox"/> |
|   | Salariu     | int         | <input checked="" type="checkbox"/> |

### Tabela Intretinuti:

O extensie a tabelii de angajati, cu scopul de a identifica cat de important este ca acel angajat sa aiba anumite avantaje: orar avantajos, salariu respectabil etc.

Un angajat cu 3 copii in intretinere va fi tratat preferential deoarece are oameni ce depind de el.

Aceasta tabela foloseste o cheie straina ID\_Angajat, provenita din cheia primara a tabelii Angajati, si are rolul de a face legatura intre angajat si numarul de intretinuti.

|  | Column Name | Data Type | Allow Nulls                         |
|--|-------------|-----------|-------------------------------------|
|  | ID_Angajat  | int       | <input type="checkbox"/>            |
|  | Varsta      | int       | <input checked="" type="checkbox"/> |

### Tabela clienti :


Urmatoarea tabela pe care o voi aborda este tabela de clienti: aceasta, in mod similar cu cea a angajatilor, are o cheie prima de tip int, cu auto incrementare, cu ajutorul careia putem identifica cu usurinta orice client cu care am interactionat. Este practic o copie ideologica a tabelii

|   | Column Name | Data Type   | Allow Nulls                         |
|---|-------------|-------------|-------------------------------------|
| 🔑 | ID_Client   | int         | <input type="checkbox"/>            |
|   | Nume        | varchar(50) | <input checked="" type="checkbox"/> |
|   | Prenume     | varchar(50) | <input checked="" type="checkbox"/> |
|   | CNP         | varchar(50) | <input type="checkbox"/>            |
|   | Telefon     | varchar(50) | <input checked="" type="checkbox"/> |

Angajati, doar ca directionata catre clienti. Informatii aditionale pot fi adaugate in continuarea etapelor de proiectare, pentru a identifica cu mai mare acuratete anumite tipicuri ale clientilor.


### Tabela Comenzi

O tabela foarte importanta este tabela Comenzi. Ea leaga fiecare comanda de clientul care a efectuat comanda si de angajatul care a aprobat-o. Campurile ID\_Angajat si ID\_Client sunt chei straine, aduse din tabelele Angajati, respectiv Clienti.

|   | Column Name | Data Type | Allow Nulls                         |
|---|-------------|-----------|-------------------------------------|
|  | ID_Comanda  | int       | <input type="checkbox"/>            |
|   | ID_Angajat  | int       | <input type="checkbox"/>            |
|   | ID_Client   | int       | <input checked="" type="checkbox"/> |


### Tabela Biciclete:

Aceasta tabela este cea mai importanta din baza de date, deoarece reprezinta obiectul care urmeaza sa fie inchiriat in cadrul magazinului nostru prototipat. ID\_Bicicleta este cheie primara, din nou cu autoincrementare, si tabela mai include si campuri care sa ajute cu identificarea unei anumite biciclete, precum Producatorul, tipul bicicletei sau dimensiunea rotilor, precum si o categorie optionala de categorie de pret, deoarece bicicletele pot varia enorm in materiale si finisaj.

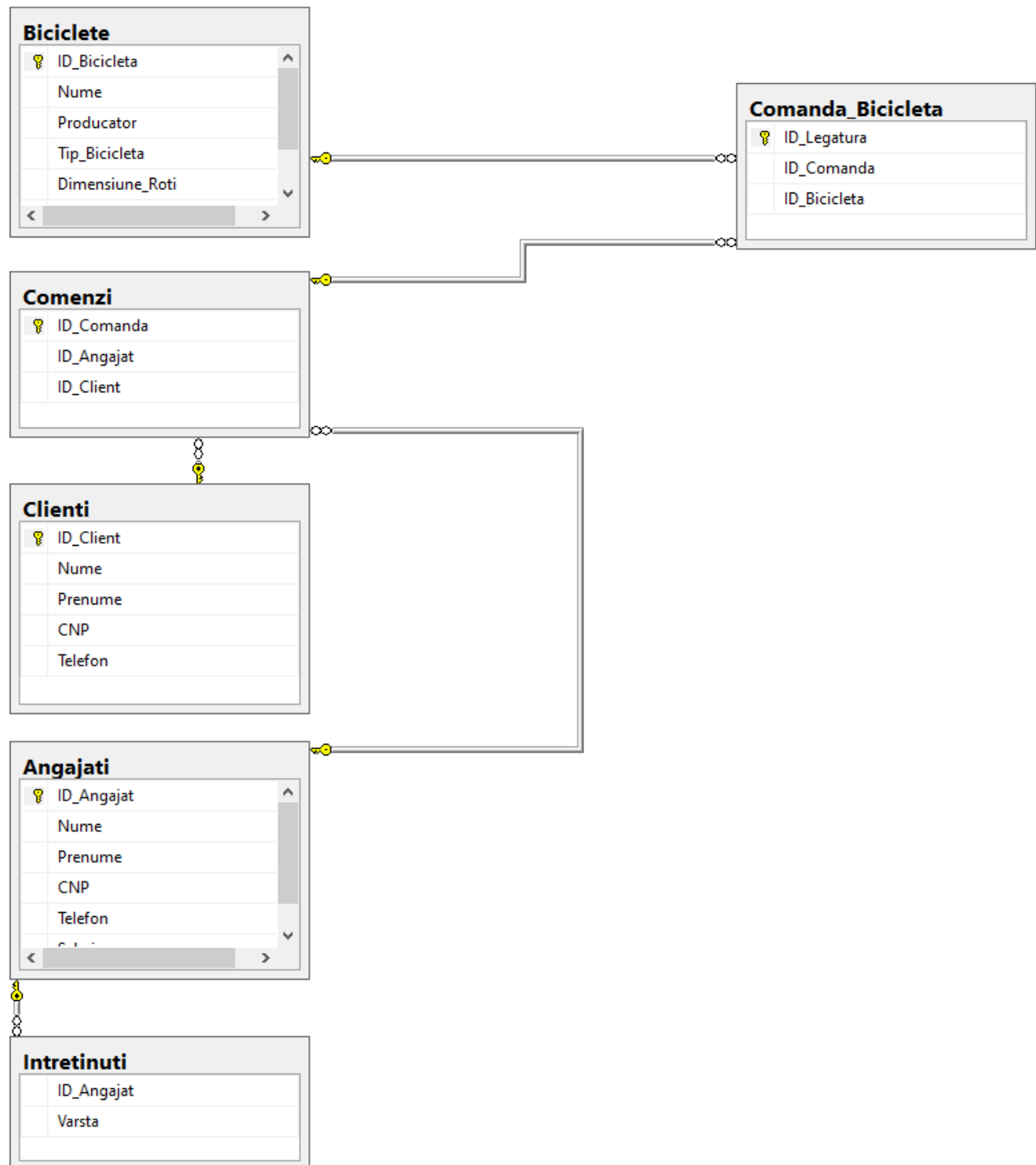
|   | Column Name     | Data Type   | Allow Nulls                         |
|---|-----------------|-------------|-------------------------------------|
|  | ID_Bicicleta    | int         | <input type="checkbox"/>            |
|   | Nume            | varchar(50) | <input checked="" type="checkbox"/> |
|   | Producator      | varchar(50) | <input type="checkbox"/>            |
|   | Tip_Bicicleta   | varchar(50) | <input type="checkbox"/>            |
|   | Dimensiune_Roti | int         | <input checked="" type="checkbox"/> |
|   | Categorie_Pret  | varchar(50) | <input checked="" type="checkbox"/> |
|   | Pret            | int         | <input checked="" type="checkbox"/> |

### Tabela Comanda\_Bicicleta:

Aceasta tabela face legatura dintre Comanda si Bicicleta. Ea este necesara pentru a putea realiza o relatie N:N intre comanda si bicicleta.

|   | Column Name  | Data Type | Allow Nulls                         |
|---|--------------|-----------|-------------------------------------|
|  | ID_Legatura  | int       | <input type="checkbox"/>            |
|   | ID_Comanda   | int       | <input checked="" type="checkbox"/> |
|   | ID_Bicicleta | int       | <input type="checkbox"/>            |

## Relatia dintre tabele



Relatia Many-Many: Am considerat necesar sa implementez o relatie N:N intre Biciete si Comenzi, in sensul ca o comanda poate avea mai multe biciclete, si o bicicleta poate fi continuta in mai multe comenzi. Acest lucru l-am implementat cu o tabelade legatura care contine ID-urile comenzii, a bicicletei si un ID al legaturii.

Celelalte relatii sunt de tipul 1:N si exista intre Clienti-Comenzi, Angajati-Comenzi si Angajati-Intretinuti. Aceste relatii ilustreaza faptul ca nu se poate ca o comanda sa fie plasata de mai multi Angajati, sau ca mai multi clienti pot fi titulari ai aceleiasi comenzi, si in acelasi timp semnifica faptul ca un client poate plasa mai multe comenzi, ca un angajat poate superviza mai multe comenzi, si poate avea mai multi intretinuti.

## Interfata

Interfata contine toate elementele de interactiune cu baza de date necesare. Afisare, adaugare, sterere precum si o serie de elemente statistice menite sa scoata la iveala cele mai semnificative performante.

In principiu fiecare dintre butoanele de pe interfata este destul de usor de anticipat ca utilitate, cu o potentiala exceptie la zona din dreapta jos a plasarii comenzii, pe care am impartit-o in doua etape deoarece trebuie sa fac aditii in doua tabele separate. Este necesar ca prima oara sa se apese pe butonul intitulat Etapa 1 deoarece acesta face modificarile necesare in tablea Comenzi, iar cu informatia comenzii plasate, Etapa 2 poate face legatura intre Comanda si Bicicleta adaugand id-ul bicicletei care a fost inclusa in acea comanda.

Interogari:

Vom aborda interogariile pe categorii:

Prima categorie este cea a Afisarilor pentru tabele, si sunt extrem de simple :

`Select Nume, Prenume, CNP, Telefon, Salariu from dbo.Angajati` – pentru tabela angajati

`Select Nume, Prenume, CNP, Telefon from dbo.Clienti` -pentru Clienti

`Select Nume, Producator, Tip_Bicicleta, Pret from dbo.Biciclete` -pentru biciclete

Mai exista si interogari pentru cautarea bicicletelor sau a unui angajat anume. Din nou, acestea nu prezinta o dificultate semnificativa:

`Select Nume, Producator, Dimensiune_Roti, Categorie_Pret, Pret from dbo.Biciclete where Tip_Bicicleta=@tip`

Unde @tip este variabila care va fi preluata din text box.

`Select Nume, Prenume, CNP, Telefon from dbo.Angajati where CNP=@cnp` si aceasta interogare care afiseaza angajatii cu CNP-ul specificat in text box.

Cea de-a doua categorie ar fi cea care manageriaza baza de date: deci operatii precum Insert, Update si Delete:

`INSERT INTO dbo.Angajati (Nume, Prenume, CNP, Telefon, Salariu) VALUES (@nume, @prenume, @cnp, @telefon, @salariu)` Aceasta interogare are rolul de a adauga informatiile furnizate in cele 4 text box-uri de sub „Manageriere Angajati” in tabela angajati.

Si omologul sau pentru situatia posibila in care un angajat isi schimba numarul de telefon: `Update dbo.Angajati Set Telefon=@telefon Where CNP=@cnp`

Absolut similar este procedeul si pentru Clienti, mai ales datorita faptului ca aceasta tabela are acelasi format de date ca si Angajati: `INSERT INTO dbo.Clienti (Nume, Prenume, CNP, Telefon) VALUES (@nume, @prenume, @cnp, @telefon)` si `Update dbo.Clienti Set Telefon=@telefon Where CNP=@cnp`

Pentru tabela intretinuti, trebuie sa adaugam in tabela ID-ul angajatului si varsta intretinutului, cu ajutorul unui CNP de angajat( deoarece nu putem lucra cu ID-uri direct)

```
INSERT INTO dbo.Intretinuti (ID_Angajat, Varsta) SELECT A.ID_Angajat, @varsta FROM Angajati A
WHERE A.CNP= @cnp
```

In ceea ce priveste tabela Angajati, am considerat necesar sa avem si o functionalitate pentru stergerea angajatului, in cazul in care acesta nu mai lucreaza in cadrul companiei. Asadar putem sa efectuam stergerea, dupa ce facem o cautare pe baza CNP-ului( am ales CNP pentru ca este unic si nu pot exista coincidente) : `Delete dbo.Angajati where CNP=@cnp`

A treia categorie este cea a interogarilor cu JOIN( cele mai simple):

Prima interogare are ca scop realizarea unei ordonari a angajatilor in functie de numarul de comenzi plasate, si arata cam asa:

```
SELECT A.Nume, A.Prenume, Count(C.ID_Angajat)
FROM Angajati A Left JOIN Comenzi C ON A.ID_Angajat = C.ID_Angajat
GROUP BY A.Nume, A.Prenume
ORDER BY Count(C.ID_Angajat) DESC
```

Adica grupam tabela Angajati cu tabela Comenzi pentru a putea numara de cate ori apare fiecare angajat in acea tabela, si la final ordonam descrescator dupa numarul de aparitii.

Cea de-a doua interogare cu join doreste sa selecteze bicicletele care au fost inchiriate de mai mult de doua ori, pentru a reusi sa observam care sunt bicicletele cele mai dorite de catre clienti:

```
SELECT B.Producator, B.Tip_Bicicleta, B.Nume, COUNT(CB.ID_Bicicleta)
FROM Biciclete B left join Comanda_Bicicleta CB on B.ID_Bicicleta = CB.ID_Bicicleta
Group by B.Producator, B.Tip_Bicicleta, B.Nume
HAVING COUNT(CB.ID_Bicicleta) >= 2
ORDER BY COUNT(CB.ID_Bicicleta) DESC"
```

Similar din punct de vedere ideologic cu interogarea precedenta, trebuie sa extragem bicicletele cele mai inchiriate, iar acest lucru presupune in primul rand sa numaram de cate ori apare fiecare bicicleta in comenzi, informatie ce se afla in tabela Comanda\_Bicicleta. Asadar este clar ca ne trebuie un JOIN intre aceste doua tabele, si acest join se poate face pe conditia identitatii dintre ID\_Bicicleta din cele doua tabele. Odata ce am reusit sa numaram de cate ori apare fiecare bicicleta, am mai impus conditia ca bicicleta sa fie inchirata de cel putin doua ori, si am ordinar aceasta lista descrescator.

A treia interogare presupune aflarea clientilor care au fost suficient de multumiti de serviciile noastre pentru a se reintoarce:

```
SELECT C.Nume, C.Prenume, Count(Co.ID_Client)
FROM Clienti C JOIN Comenzi Co ON C.ID_Client = Co.ID_Client
GROUP BY C.Nume, C.Prenume
Having Count(Co.ID_Client) > 1
```

Deci daca gasim acelasi ID de client de mai mult de o singura data, asta inseamna ca acel client a ales sa revina la noi.

Urmatoarea interogare vizeaza aflarea angajatilor care au cel mai mare numar de intretinuti, acest lucru fiind important deoarece mai multe persoane depind de venitul acestuia.

```
SELECT A.Nume, A.Prenume, Count(I.ID_Angajat)
FROM Angajati A Left JOIN Intretinuti I ON A.ID_Angajat = I.ID_Angajat
GROUP BY A.Nume, A.Prenume
ORDER BY Count(I.ID_Angajat) DESC
```

Aceasta tabela este structurata ca o lista formata din id-ul angajatului si varsta intretinutului de catre acesta. Asadar o numarare a datilor cand apare acesta in lista ne va da numarul de intretinuti avuti de acesta.

Query-ul care urmeaza returneaza angajatul care a plasat comanda pentru clientul X, unde X este parametru venit din interfata, dintr-un text box:

```
SELECT A.Nume, A.Prenume
FROM Angajati A JOIN Comenzi C ON A.ID_Angajat = C.ID_Angajat
WHERE C.ID_Client =(Select ID_Client FROM Clienti WHERE CNP= @cnp)
```

Deci cautam acel angajat care a plasat comanda pentru clientul al carui CNP se potriveste cu cel care a fost introdus in text box.

Ultima interogare din acest set este cea care determina angajatii care au plasat comenzi pentru cea mai mare valoare de biciclete. Pe scurt: cine a fost cel mai de succes angajat

```
SELECT A.Nume, A.Prenume, SUM(B.Pret)
FROM Angajati A JOIN Comenzi C on A.ID_Angajat = C.ID_Angajat JOIN Comanda_Bicicleta CB on
C.ID_Comanda = CB.ID_Comanda JOIN Biciclete B on CB.ID_Bicicleta = B.ID_Bicicleta
Group by A.Nume, A.Prenume
```

Ultima Pentru aceasta interogare trebuie sa facem legaturile dintre Angajati si Comenzi precum si legatura cu tabela Comanda\_Bicicleta, iar odata ce avem aceasta legatura facuta, putem adauga sumele pentru fiecare bicicleta si sa obtinem un total pentru fiecare angajat.

In continuare voi aborda putin modalitate de plasare a unei comenzi, deoarece este interesanta. In baza mea de date am o tabela de comenzi care ne spune ce client a plasat o comanda si ce angajat a acceptat acea comanda, si pe langa aceasta mai este si tabela Comanda\_Bicicleta care ne spune ce comanda a fost plasata pentru ce bicicleta. Asadar procesul nostru de plasare a unei comenzi se face in doi pasi :

Etapa 1- presupune creerea comenzii in tabela Comenzi. Deci ne trebuie cnp pentru angajat si client pentru a putea efectua cate o cautare in fiecare dintre tabelele lor si a scoate id-ul care trebuie plasat in tabela:

```
INSERT INTO dbo.Comenzi (ID_Angajat, ID_Client) SELECT ID_Angajat, ID_Client FROM Angajati A Cross
JOIN Clienti C Where A.CNP = @cnpangajat AND C.CNP = @cnpclient
```

Am ales Cross Join pentru a putea avea o selectie pe baza a cnp angajat si cnp client simultan, si nu m-am putut gandi la o alta modalitate de a lega doua tabele de acest fel. Outer join sau Union nu mergeau( sau nu am reusit eu)

Etapa 2 are in vedere adaugarea legaturii dintre comanda plasata si bicicleta inclusa in acea comanda. Asadar, vom lua ultima comanda introdusa in Comenzi si o vom pune in Comanda\_Bicicleta impreuna cu id-ul bicicletei care corespunde cu numele introdus in campul destinat:

```
INSERT INTO dbo.Comanda_Bicicleta (ID_Comanda, ID_Bicicleta) SELECT ID_Comanda, ID_Bicicleta From
Comenzi C CROSS Join Biciclete B WHERE ID_Comanda = (Select Top 1 C.ID_Comanda FROM Comenzi C Order by
C.ID_Comanda DESC) AND Nume = @nume"
```

Ultima categorie este reprezentata de interogările cu subcereri:

Prima interogare are ca scop afisarea angajatilor cu un salariu peste media companiei

```
SELECT A.Nume, A.Prenume, A.Salariu
FROM Angajati A GROUP BY A.Nume, A.Prenume, A.Salariu
HAVING A.Salariu > (SELECT AVG(Salariu)
                    FROM Angajati)
```

A doua interogare are ca scop afisarea bicicletelor care au un pret peste cel mediu al categoriei din care fac parte(Mountain/ Road)

```
SELECT B.Nume, B.Pret
FROM Biciclete B
Where B.Tip_Bicicleta = @tip
GROUP BY B.Nume, B.Pret
Having B.Pret > (Select AVG(Pret)
                From Biciclete
                Where Tip_Bicicleta = @tip)
```

In al treilea rand avem o interogare al carei scop este sa afiseze primii 5 angajati in ordinea salariilor:

```
SELECT Ang.Nume, Ang.Prenume, Ang.Salariu
FROM(SELECT TOP 5 A.Nume, A.Prenume, A.Salariu
      FROM Angajati A
      ORDER BY A.Salariu DESC) AS Ang
```

Urmatorul query trebuie sa afiseze acei clienti care au inchiriat mai multe biciclete in acelasi timp, acest lucru fiind important in caz ca dorim sa oferim promotii de grup.

```
SELECT CL.Nume, CL.Prenume, CL.CNP
FROM Clienti CL
WHERE CL.ID_Client IN(SELECT C.ID_Client
```



```
FROM Comenzi C
WHERE C.ID_Comanda IN (SELECT CB.ID_Comanda
                        FROM Comanda_Bicicleta CB
                        GROUP BY CB.ID_Comanda
                        Having COUNT(CB.ID_Bicicleta) > 1))"
```

Ultimul query afiseaza topul bicicletelor ca incasari, pentru a vedea care dintre biciclete are cel mai mare succes comercial

```
SELECT b.ID_Bicicleta, b.Nume, b.Pret*cb.count as Incasari
FROM(SELECT B.ID_Bicicleta, B.Pret, B.Nume
      FROM Biciclete B) b JOIN(SELECT CB.ID_Bicicleta, COUNT(CB.ID_Bicicleta) AS count
                                FROM Comanda_Bicicleta CB
                                GROUP BY CB.ID_Bicicleta) cb ON b.ID_Bicicleta = cb.ID_Bicicleta
ORDER BY Incasari DESC";
```