

Proiect Baze de Date

Caraeane Bogdan-Andrei

Cuprins

1.Descrierea modelului real, a utilităţii şi a regulilor de funcţionare	3
2.Prezentarea constrângerilor (restricţii, reguli)	4
3.Descrierea entităţilor şi precizarea cheii primare	5
4.Descrierea relaţiilor şi precizarea cardinalităţii	7
5.Descrierea atributelor, tipuri şi restricţii	8
6. Realizarea diagramei entitate-relaţie	16
7. Realizarea diagramei conceptuale	17
8. Schemele relaţionale generate din diagrama conceptuală	17
9. Realizarea normalizării până la forma normală 3 (FN1-FN3).	18
10.Crearea unei secvenţe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).	20
11. Crearea tabelelor în SQL şi inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative; maxim 30 de înregistrări în fiecare tabel).	21
11.1 Creearea tabelelor SQL	21
11.2 Inserarea datelor in tabele	25
12. Cereri SQL complexe	41
12.1 Coduri SQL pe tipuri de cerere	41
1. Subcerere sincronizata in SELECT (subquery corelata pe 3 tabele) . . .	41
2. Subcerere nesincronizata in FROM (derived table)	42
3. Grupare si filtrare pe grupuri cu HAVING si subquery	43
4. Ordonari si functii NVL, DECODE	44
5. Bloc WITH, functii pe siruri, functii pe date calendaristice, CASE . . .	45
13. Operaţii de actualizare şi suprimare folosind subcereri	47
13.1 Operaţii de actualizare (UPDATE)	47

13.2 Operații de suprimare (DELETE)	48
14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.	49
Exemplu de operație LMD permisă (ex: UPDATE)	49
Exemplu de operație LMD nepermisă pe vizualizarea complexă	50
15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația <i>outer-join</i> pe minimum 4 tabele, o cerere ce utilizează operația <i>division</i> și o cerere care implementează analiza <i>top-n</i>.	50
a) Outer Join pe minimum 4 tabele	50
b) Operația de DIVISION	50
c) Analiza TOP-N	51
16. Optimizarea și analiza execuției unei cereri SQL	51
a) Optimizarea unei cereri utilizând proprietățile algebrei relaționale	51
a.i. Cerere inițială	51
a.ii. Cerere optimizată	52
b) Prezentarea planului de execuție și optimizarea performanței	53
b.i. Cerere SQL de analiză	53
b.ii. Plan de execuție fără optimizare	53
b.iii. Optimizare cu index și hint Oracle	54
b.iv. Plan de execuție optimizat	54
b.v. Comparatie între execuții	54
17. Normalizare avansată și denormalizare	54
a. Realizarea normalizării BCNF, FN4, FN5	54
Exemplu non-BCNF și normalizare la BCNF	54
Exemplu non-FN4 și normalizare la FN4	55
Exemplu non-FN5 și normalizare la FN5	55
b. Aplicarea denormalizării și justificarea necesității	56
Exemplu de denormalizare și justificare	56

1.Descrierea modelului real, a utilității și a regulilor de funcționare

O aplicație pentru rezervarea spațiilor de cazare într-o stațiune de schi oferă utilizatorilor (turiștilor) posibilitatea să caute un spațiu disponibil într-una dintre cabanele companiei pentru perioade determinate. Fiecare schior, identificat printr-un număr unic de înregistrare, va avea completată o fișă care va cuprinde detalii privind experiența de schi, nevoi specifice de echipament și orice alte cerințe suplimentare (de exemplu, lecții de schi, închiriere de echipament etc.). De asemenea, utilizatorii vor completa obligatoriu date de contact, cum ar fi număr de telefon și adresă de email.

Biletul reprezintă documentul de acces al unui schior la un eveniment tematic din stațiune și include: un identificator unic, prețul biletului, data evenimentului și legături către schior (deținător) și eveniment (detaliile sale).

Abonamentul conferă schiorului acces nelimitat la toate ascensoarele din stațiune și conține pretul și perioada platită, iar fiecare acces la un ascensor este înregistrat în entitatea `Intrare_Ascensor` pentru a urmări exact unde și când a folosit instalația.

Rezervarea electronică din cadrul aplicației stațiunii de schi se concentrează exclusiv pe asigurarea echipamentului de schi și a spațiului de cazare. Astfel, turiștii pot selecta cu ușurință unitatea de cazare dorită fie aceasta interioară, fie exterioară în funcție de preferințele și necesitățile lor. În același timp, se poate efectua rezervarea echipamentului de schi, permițând utilizatorilor să aleagă dintr-o varietate de opțiuni adaptate nivelului lor de experiență.

Tarifele pentru cazare sunt exprimate ca preț unitar pe zi și variază în funcție de tipul de camera.

Tarifele pentru închirierea echipamentului de schi sunt exprimate ca preț unitar pe zi și sunt stabilite în funcție de tipul și calitatea echipamentului (schiuri, clăpări, căști etc.).

2. Prezentarea constrângerilor (restricții, reguli)

- Fiecare cameră aparține exact unei singure cabane.
- cabană trebuie să fie localizată într-o singură stațiune și fiecare stațiune poate avea mai multe cabane.
- Fiecare cabana are nume unic.
- Fiecare cameră are 1–4 paturi (minim 1, maxim 4) și poate caza între 1 și 4 persoane.
- `Tip_Camera` nu poate exista fără cel puțin o cameră asociată.
- Fiecare tip de cameră trebuie să aibă definit minimum un `tarif_zi`.
- rezervare de cameră trebuie să includă cel puțin o cameră și maximum 5 camere per rezervare.
- cameră nu poate fi rezervată de două ori pentru intervale de timp care se suprapun.
- Data de sosire a unei rezervări trebuie să fie cel puțin ziua curentă și maximum la 6 luni în viitor.
- Durata unei rezervări de cameră trebuie să fie de minim 1 noapte și maxi 14 nopți.
- închiriere de echipament trebuie să aibă perioada de utilizare între 1–14 zile.
- Echipamentele de calitate “premium” trebuie să aibă garanție/mentenanță valabilă minimum 2 ani.
- Un echipament nu poate fi închiriat de două ori pentru intervale de timp care se suprapun.
- Stocurile de echipament (număr de exemplare disponibile) trebuie să fie actualizate după fiecare închiriere și nu pot deveni negative.
- Un schior nu poate avea mai multe echipamente în aceeași perioadă.
- Un schior nu poate avea două abonamente care se suprapun în intervalul de valabilitate.

- Fiecare abonament la ascensor trebuie să aibă perioada de valabilitate între 1 zile (minim) și 3 luni (maxim).
- Intrările în ascensor sunt valabile doar între orele de program 7: 30 - 16: 30.
- Numărul de intrări într-un singur ascensor de către același abonament într-o zi este nelimitat, dar fiecare intrare este unică, iar următoarea intrare e valabilă peste 5 minute.
- Un bilet de eveniment nu poate fi emis după data_eventiment.
- Un schior poate cumpăra maximum 1 bilet per același eveniment.
- Fiecare eveniment trebuie să poarte un nume unic în cadrul stațiunii și să aibă o dată programată și o locație.
- Fiecare eveniment are o capacitate maximă de participanți și nu pot fi emise bilete peste acest prag.
- Un eveniment este recurent
- Fiecare lecție de schi trebuie să aibă un instructor și cel puțin un cursant.
- Un instructor nu poate susține mai mult de 1 lecție simultan într-un interval orar.
- Lecțiile de schi trebuie programate cu minim 24 de ore înainte de data și ora începerii.
- Angajatul (supertip) trebuie să aparțină exact unui subtip: Cabana, Eveniment sau Ascensor.
- Cabana, evenimentul sau ascensorul trebuie să aibă minim un angajat.
- Toate prețurile (tarif_zi, pret, tarif_total) trebuie să fie strict pozitive (> 0).
- Perioadele (data_plecare data_sosire, data_expirare data_emitere) sunt obligatorii și valide.
- Anulările făcute cu mai puțin de 24 h înainte de sosire pot genera o penalizare (ex: 50% tarif).
- Fiecare rezervare de echipament include $\text{tarif_total} = \text{tarif_zi} \times \text{perioada_zile}$ și $\text{perioada_zile} > 0$.
- Un schior trebuie să fie înregistrat înainte de a realiza orice rezervare, închiriere, abonament sau achiziție de bilet.

3.Descrierea entităților și precizarea cheii primare

Entitate	Cheie primară	Observații
Partie	#id_partie	Traseu de schi din stațiune
Cabană	#id_cabana	Unitate de cazare; aparține unei singure stațiuni; are 3 camere
Cameră	#id_camera	Aparține exact unei cabane; are 1-4 paturi și capacitate 1-4 persoane

Entitate	Cheie primară	Observații
Tip_Cameră	#id_tip_camera	Categorii de camere (ex. dublă, triplă, apartament); nu poate exista fără cel puțin o cameră asociată; tarif_zi
Rezervare_Cameră	#id_rezervare_camera	Include 1–5 camere; sosire azi și 6 luni; durată 1–14 nopți
Rezervare	#id_rezervare	Asociază rezervarea camera cu camerele rezervate
Schior	#id_schior	Client înregistrat; poate rezerva camere, echipamente, abonamente și bilete și să participe la lecții
Abonament	#id_abonament	Acces la ascensor; valabil 1–90 zile; fără suprapuneri pe același schior
Ascensor	#id_ascensor	Lift de schi; program 07:30–16:30; deservește una sau mai multe pârtii
Intrare_Ascensor	#id_intrare	Înregistrare unică de folosire a ascensorului; nelimitată/zi per abonament la un interval de 5 min
Tip_Calitate	#id_tip_calitate	Categorie echipament (standard, superior, premium, copii); premium necesită
Echipament	#id echipament	garanție/mentenanță 2 ani; Model de echipament de închiriat;
Rezervare_Echipament	#id_rezervare echipament	Rezervare echipament; perioadă 1–14 zile; tarif_total = tarif_zi × zile
Inchiriere	#id_inchiriere	Alocarea unui echipament pentru fiecare rezervare, trebuie să aibă minim 1 și maxim 5.
Eveniment	#id_eveniment	Activitate specială; nume unic în stațiune; dată azi; locație
Bilet	#id_bilet	Eliberat înainte de data_eveniment; max. 1 per schior per eveniment
Instructor	#id_instructor	Conduce lecții de schi; nu poate susține 2 lecții simultan
Lecție_Schi	#id_lectie	Lecție programată 24 h înainte; minim 1 cursant; are instructor și durată orară
Angajat	#id_angajat	Supertip; aparține exact unui subtip (Cabana, Eveniment sau Ascensor)

Entitate	Cheie primară	Observații
Angajat_Cabană	#id_angajat_cabana	Lucrează în cabană; fiecare cabană are 1 angajat
Angajat_Eveniment	#id_angajat_eveniment	Lucrează la eveniment; fiecare eveniment are 1 angajat
Angajat_Ascensor	#id_angajat_ascensor	Lucrează la ascensor; fiecare ascensor are 1 angajat

4.Descrierea relațiilor și precizarea cardinalității

RELATIE	CARDINALITATE	OBSERVAȚII
Ascensor deservește Partie	one to many (Ascensor-Partie)	O pârtie nu poate exista fără un ascensor; un ascensor poate sa aiba 0 sau mai multe partii
Partie găzduiește Cabana	one to many (Partie-Cabana)	O pârtie poate exista fără cabane; o cabană trebuie să fie amplasată pe o pârtie pentru a exista.
Cabana conține Camera	one to many (Cabana-Camera, min 3)	O cabană trebuie să aibă cel puțin 3 camere, camera trebuie să aparțină unei cabane pentru a exista.
Camera are Tip_camera	many to one (Camera-Tip_camera)	Un tip de cameră trebuie să fie folosit de cel puțin o cameră; o cameră nu poate exista fără un tip asociat.
Echipament are Tip_calitate	many to one (Echipament-Tip_calitate)	Un tip de calitate nu trebuie sa aiba un echipament asociat; un echipament trebuie să aibă un tip de calitate pentru a exista.
Cabana găzduiește Eveniment	one to many (Cabana-Eveniment)	O cabană poate exista fără evenimente; un eveniment trebuie să fie găzduit de o cabană pentru a exista.
Eveniment are Angajat_eveniment	one to many (Eveniment-Angajat_eveniment)	Un eveniment trebuie să aibă cel puțin un angajat; un angajat_eveniment trebuie să fie alocat unui eveniment.
Cabana angajează Angajat_cabana	one to many (Cabana-Angajat_cabana)	O cabană trebuie să aibă cel puțin un angajat; un angajat_cabana trebuie să fie asignat unei cabane pentru a exista.
Ascensor este operat de Angajat_ascensor	one to many (Ascensor-Angajat_ascensor)	Un ascensor trebuie să aibă cel puțin un operator; un angajat_ascensor trebuie să fie repartizat la un ascensor pentru a exista.

RELATIE	CARDINALITATE	OBSERVAȚII
Schior cumpara Abonament	one to many (Schior–Abonament)	Un schior poate exista fără abonamente; un abonament trebuie să aparțină unui schior pentru a exista.
Abonament oferă acces la Ascensor	many to many (Abonament–Ascensor)	Un abonament poate exista fără ascensoare; un ascensor trebuie să fie parte din cel puțin un abonament pentru a exista.
Schior plasează Rezervare_echipament sau rezervare_camera	one to many (Schior–Rezervare)	Un schior poate exista fără rezervări; o rezervare trebuie să fie asociată unui schior pentru a exista.
Rezervare_camera include camera	Many to many (Rezervare–Camera)	O rezervare_camera nu poate exista fără camere; o cameră poate exista fără să fie rezervată vreodată.
Rezervare_echipament include Echipament	many to many (Rezervare–Echipament)	O rezervare_echipament nu poate exista fără echipamente; un echipament poate exista fără să fie rezervat.
Eveniment emite Bilet	one to many (Eveniment–Bilet)	Un eveniment nu poate exista fără bilete emise; un bilet trebuie să fie
Schior achiziționează bilet	one to many (Schior–Bilet)	Un schior poate exista fără bilet, dar un bilet trebuie să fie achiziționat de un schior
Schior ia lecții de la Instructor	many to many (Schior–Instructor)	Un schior poate exista fără lecții; un instructor trebuie să aibă cel puțin un schior la lecții

5.Descrierea atributelor, tipuri și restricții

ENTITATE: Partie

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_partie	NUMBER	13	PK	
nume	VARCHAR2	100	ex.: “Pârția Neagră”	
dificultate	VARCHAR2 (enum)	20	‘incepator’, ‘intermediar’, ‘avansat’	NUMBER
13	PK		id_ascensor	

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
DEFAULT				
‘interme- diar’				
lungime_km	DECIMAL	5,2	> 0	
desnivel_m	INTEGER	—	0; DEFAULT 0	
deschis	BOOLEAN	—	TRUE, FALSE; DEFAULT TRUE	indică dacă pârția e deschisă

ENTITATE: Cabana

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_cabana	NUMBER	13	PK	
nume	VARCHAR2	100	ex.: “Cabana Alpin”	
locatie	VARCHAR2	255	ex.: “Valea Prahovei”	
nr_camere	INTEGER	—	3	

ENTITATE: Camera

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_camera	NUMBER	13	PK	
id_cabana	NUMBER	13	FK → Cabana(id_cabana)	
nr_paturi	INTEGER	—	1 .. 4	
capacitate_pers	INTEGER	—	1 .. 4	
id_tip_camera	NUMBER	13	FK → Tip_Camera(id_tip_camera)	

ENTITATE: Tip_Camera

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_tip_camera	NUMBER	13	PK	
denumire	VARCHAR2	50	ex.: “dublă”, “apartament”	
tarif_zi	DECIMAL	8,2	> 0	

ENTITATE: Schior

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_schior	NUMBER	13	PK	
nume	VARCHAR2	100	ex.: "Popescu Ion"	
telefon	VARCHAR2	20	ex.: "07xxxxxxxx"	
email	VARCHAR2	100	ex.: "ion@exemplu.com"; UNIQUE	
nivel_experienta	VARCHAR2 (enum)	20	'incepator', 'mediu', 'avansat' DEFAULT 'incepator'	
data_nasterii	DATE	—	< CURRENT_DATE	
varsta	INTEGER	—	0	poate fi calculată automat din data_nasterii

ENTITATE: Abonament

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_abonament	NUMBER	13	PK	
id_schior	NUMBER	13	FK → Schior(id_schior)	
data_emitere	DATE	—	CURRENT_DATE	
data_expirare	DATE	—	Data emitere plus (1 .. 90) zile	
pret	DECIMAL	8,2	> 0	

ENTITATE: Ascensor

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_ascensor	NUMBER	13	PK	
denumire	VARCHAR2	100	ex.: "Ascensor Nord"	
program_start	VARCHAR2	5	DEFAULT '07:30'	format HH24:MI
program_end	VARCHAR2	5	DEFAULT '16:30'	format HH24:MI
tip_ascensor	VARCHAR2 (enum)	20	'scaun', 'teleschi', 'gondola' DEFAULT 'scaun'	

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
deschis	BOOLEAN	—	TRUE, FALSE; DEFAULT TRUE	indică dacă funcționează

ENTITATE: Tip_Calitate

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_tip_calitate	NUMBER	13	PK	
denumire	VARCHAR2 (enum)	20	'standard', 'superior', 'premium', 'copii'	
garantie_ani	INTEGER	—	2 dacă denumire='premium'; altfel 0	

ENTITATE: Echipament

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_echipament	NUMBER	13	PK	
id_tip_calitate	NUMBER	13	FK → Tip_Calitate(id_tip_calitate)	
denumire	VARCHAR2	100	ex.: "Schi Atomic"	
stoc_total	INTEGER	—	1	
stoc_disponibil	INTEGER	—	0 .. stoc_total	

ENTITATE: Rezervare_Camera

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_rezervare_camera	NUMBER	13	PK	
id_schior	NUMBER	13	FK → Schior(id_schior)	
data_sosire	DATE	—	între CURRENT_DATE și +6 luni	
durata_nopti	INTEGER	—	1 .. 14	
nr_camere	INTEGER	—	1 .. 5	
tarif_total	DECIMAL	10,2	> 0	calculat automat

ENTITATE: Rezervare

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_rezervare	NUMBER	13	PK	
id_rezervare_camera	NUMBER	13	PK	
id_camera	NUMBER	13	FK → Cameră(id_camera)	
data_sosire	DATE	—	între CURRENT_DATE și ADD_MONTHS(CURRENT_DATE, 6)	
durata_nopti	INTEGER	—	între 1 și 14	
tarif_total	DECIMAL	10,2	> 0	calculat ca (tarif_zi pentru fiecare cameră) × nopti

ENTITATE: Rezervare_Echipament

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_rezervare_echipament	NUMBER	13	PK	
id_schior	NUMBER	13	FK → Schior(id_schior)	
data_inceput	DATE	—	orice dată	
data_final	DATE	—	data_inceput .. max 14 zile	
tarif_total	DECIMAL	10,2	> 0	calculat automat
Numar_echipamente	INTEGER		1	

ENTITATE: Închiriere

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_inchiriere	NUMBER	13	PK	
id_rezervare_echipament	NUMBER	13	FK	
id_echipament	NUMBER	13	FK → Echipa- ment(id_echipament)	
data_inceput	DATE	—	CURRENT_DATE	
data_final	DATE	—	între data_inceput și data_inceput + INTERVAL '14' DAY	

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
tarif_total	DECIMAL	10,2	> 0	calculat ca tarif_zi × nu- măr_zile
observații_opționale	VARCHAR2	255	—	ex. “bocanci mărime 42, cască”

ENTITATE: Eveniment

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_eveniment	NUMBER	13	PK	
nume	VARCHAR2	100	ex.: “Concert Après-ski”	
data_eveniment	DATE	—	CURRENT_DATE	
id_cabana	NUMBER	13	FK → Cabana(id_cabana)	locatia eve- nimentului capaci- tate_max
INTEGER	—	1		

ENTITATE: Bilet_Eveniment

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_bilet	NUMBER	13	PK	
id_schior	NUMBER	13	FK → Schior(id_schior)	
id_eveniment	NUMBER	13	FK → Eveniment(id_eveniment)	
data_emitere	DATE	—	data_eveniment	
pret	DECIMAL	8,2	> 0	

ENTITATE: Instructor

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_instructor	NUMBER	13	PK	
nume	VARCHAR2	100	text liber	
specializare	VARCHAR2	100	ex.: “freestyle”, “alpina”	

RELATIE: Lectie_Schi

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_lectie	NUMBER	13	PK	
id_instructor	NUMBER	13	FK → Instruc- tor(id_instructor)	
id_schior	NUMBER	13	FK → Schior(id_schior)	
data_lectie	DATE	—	SYSDATE + INTERVAL '24' HOUR	programare minim 24 h înainte
durata_ore	INTEGER	—	> 0	

ENTITATE: Angajat

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_angajat	NUMBER	13	PK	
nume	VARCHAR2	100	text liber	
tip	VARCHAR2 (enum)	20	'cabana', 'eveniment', 'ascensor'	

RELATIE: Angajat_Cabana

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_angajat_cabana	NUMBER	13	PK	
id_angajat	NUMBER	13	FK → Angajat(id_angajat)	
id_cabana	NUMBER	13	FK → Cabana(id_cabana)	

RELATIE: Angajat_Eveniment

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_angajat_eveniment	NUMBER	13	PK	
id_angajat	NUMBER	13	FK → Angajat(id_angajat)	
id_eveniment	NUMBER	13	FK → Eveni- ment(id_eveniment)	

RELATIE: Angajat_Ascensor

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_angajat_ascensor	NUMBER	13	PK	
id_angajat	NUMBER	13	FK → Angajat(id_angajat)	
id_ascensor	NUMBER	13	FK → Ascensor(id_ascensor)	

ENTITATE: Intrare_Ascensor

Atribut	Tip de date	Dimensiune / Precizie	Valori posibile / implicite	Observații
id_intrare	NUMBER	13	PK	
id_abonament	NUMBER	13	FK → Abonament(id_abonament)	
id_ascensor	NUMBER	13	FK → Ascensor(id_ascensor)	
data_intrare	DATE	—	între “07:30” și “16:30”	acces valid doar în program

6. Realizarea diagramei entitate-relație

Diagrama ER este oferită în anexa grafică (Figura 1) și prezintă toate entitățile, atributele și relațiile cu cardinalități și constrângeri.

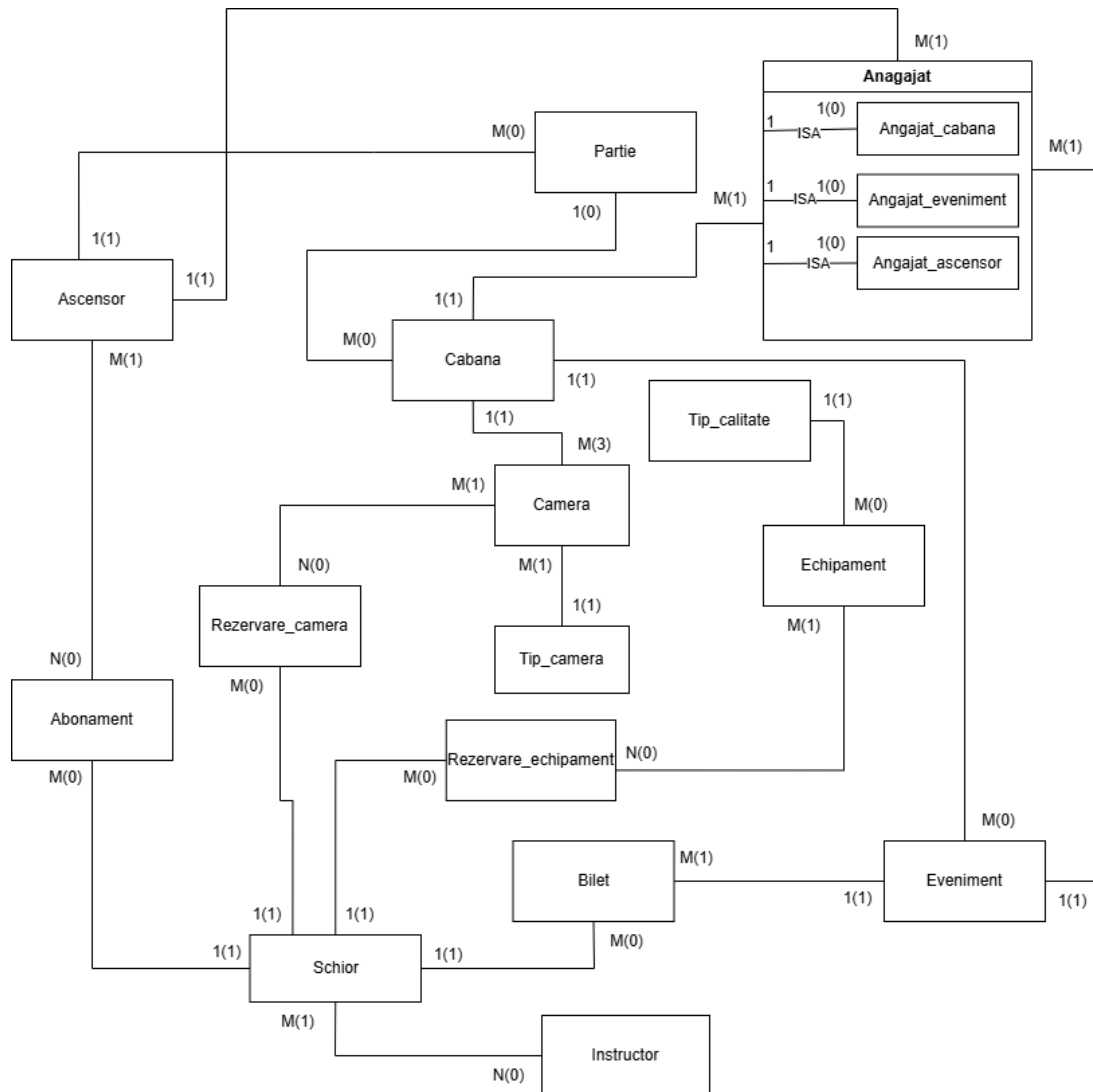


Figura 1: Diagrama entitate-relație

7. Realizarea diagramei conceptuale

Diagrama conceptuală derivată include cel puțin 7 tabele independente și entități asociative (Figura 2). Reflectă structura relațională în FN3.



Figura 2: Diagrama conceptuală

8. Schemele relaționale generate din diagrama conceptuală

- **PARTIE**(#id_partie, id_ascensor, denumire, dificultate, lungime_km, desnivel_m, deschis)
- **CABANĂ**(#id_cabana, nume, locatie, id_partie, nr_camere)
- **CAMERĂ**(#id_camera, id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
- **TIP_CAMERA**(#id_tip_camera, denumire, tarif_zi)
- **TIP_CALITATE**(#id_tip_calitate, denumire, garantie_ani)

- **ECHIPAMENT**(#id_echipament, id_tip_calitate, denumire, stoc_total, stoc_disponibil)
- **SCHIOR**(#id_schior, nume, telefon, email, nivel_experienta, data_nasterii)
- **ABONAMENT**(#id_abonament, id_schior, data_emitere, data_expirare, pret)
- **ASCENSOR**(#id_ascensor, denumire, program_start, program_end, tip_ascensor, deschis)
- **EVENIMENT**(#id_eveniment, nume, data_eveniment, id_cabana, capacitate_max)
- **BILET**(#id_bilet, id_eveniment, id_schior, data_inchiriere, data_inchiriere_final, numar_echipamente, tarif_total)
- **INSTRUCTOR**(#id_instructor, nume, specializare)
- **LECTIE_SCHI**(#id_lectie, id_instructor, id_schior, data_lectie, durata_ore)
- **REZERVARE_CAMERA**(#id_rezervare_camera, id_schior, data_sosire, durata_nopti, nr_camere, tarif_total)
- **REZERVARE**(#id_rezervare, #id_rezervare_camera, #id_camera, data_sosire, durata_nopti, tarif_total)
- **REZERVARE_ECHIPAMENT**(#id_rezervare_echipament, id_schior, data_inceput, data_final, numar_echipamente, tarif_total)
- **INCHIRIERE**(#id_inchiriere, #id_rezervare_echipament, #id_echipament, data_inceput, data_final, tarif_total, observatii_optionale)
- **ANGAJAT**(#id_angajat, nume, tip)
- **ANGAJAT_CABANA**(#id_angajat_cabana, id_angajat, id_cabana)
- **ANGAJAT_EVENIMENT**(#id_angajat_eveniment, id_angajat, id_eveniment)
- **ANGAJAT_ASCENSOR**(#id_angajat_ascensor, id_angajat, id_ascensor)
- **INTRARE_ASCENSOR**(#id_intrare, id_abonament, id_ascensor, data_intrare)

9. Realizarea normalizării până la forma normală 3 (FN1-FN3).

În această secțiune prezentăm trei cazuri din modelul aplicației de rezervări la stațiune schi: un exemplu non-FN1, un exemplu non-FN2 și un exemplu non-FN3, împreună cu transformările necesare.

Exemplu A: relația SCHIOR (non-FN1)

Relația inițială (non-FN1):

```
SCHIOR(id_schior, nume, telefon, emailuri, nivel_experienta,
      data_nasterii)
```

unde atributul **emailuri** este multivaloare (un schior poate avea mai multe adrese de email de contact).

Aducere în FN1:

Se extrage atributul multivaloare într-o relație separată:

SCHIOR_EMAIL(id_schior, email)

și se elimină emailuri din SCHIOR, obținându-se:

SCHIOR(id_schior, nume, telefon, nivel_experienta, data_nasterii)

Verificare FN2 și FN3:

SCHIOR are cheia candidat {id_schior} și nu există dependențe parțiale sau tranzitive, deci este în FN3.

Exemplu B: relația REZERVARE (FN1, dar non-FN2)

Relația inițială:

REZERVARE(id_rezervare, id_rezervare_camera, id_camera, data_sosire,
durata_nopti, tarif_total)

Determinări funcționale:

$$F = \{ (id_rezervare, id_rezervare_camera, id_camera) \rightarrow (data_sosire, durata_nopti, tarif_total) \\ id_rezervare_camera \rightarrow (data_sosire, durata_nopti, tarif_total) \}$$

Verificare FN2 (non-FN2):

Există dependență parțială: $id_rezervare_camera \rightarrow data_sosire, durata_nopti, tarif_total$,
unde {id_rezervare, id_rezervare_camera, id_camera} este cheia candidat.

Descompunere în FN2:

REZERVARE_CAMERA(id_rezervare_camera, data_sosire, durata_nopti,
tarif_total)
REZERVARE_F2(id_rezervare, id_rezervare_camera, id_camera)

Verificare FN3:

Ambele relații rezultate nu conțin dependențe tranzitive, deci sunt în FN3.

Exemplu C: relația ECHIPAMENT (FN1, FN2, dar non-FN3)

Relația inițială:

ECHIPAMENT(id_echipament, denumire, tip, garantie_ani)

Determinări funcționale:

$$F = \{ id_echipament \rightarrow denumire, tip, garantie_ani; \\ tip \rightarrow garantie_ani \}$$

Verificare FN3 (non-FN3):

Există dependență tranzitivă: $id_echipament \rightarrow tip \rightarrow garantie_ani$

Descompunere în FN3:

TIP_ECHIPAMENT(tip, garantie_ani)
ECHIPAMENT_F3(id_echipament, denumire, tip)

Prin aceste transformări, relațiile din modelul aplicației ajung în FN3, conform cerințelor normalizării.

10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).

Secvențe auto-increment

```
CREATE SEQUENCE seq_id_partie
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_cabana
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_camera
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_tip_camera
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_tip_calitate
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id echipament
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_schior
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_abonament
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_ascensor
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_eveniment
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_bilet
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_instructor
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_lectie
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_rezervare_camera
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_rezervare
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_rezervare_echipament
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_inchiriere
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_angajat
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_angajat_cabana
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_angajat_eveniment
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_angajat_ascensor
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
CREATE SEQUENCE seq_id_intrare
  START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
```

11. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tablele asociative; maxim 30 de înregistrări în fiecare tabel).

11.1 Crearea tabelelor SQL

Crearea tabelelor SQL

```
CREATE TABLE Ascensor (  
  id_ascensor    NUMBER DEFAULT seq_id_ascensor.NEXTVAL PRIMARY KEY,  
  denumire       VARCHAR2(100) NOT NULL,  
  program_start  VARCHAR2(5) DEFAULT '07:30'  
  CHECK (  
    SUBSTR(program_start,3,1) = ':'  
    AND LENGTH(program_start) = 5  
    AND TO_NUMBER(SUBSTR(program_start,1,2)) BETWEEN 0 AND 23  
    AND TO_NUMBER(SUBSTR(program_start,4,2)) BETWEEN 0 AND 59  
  ),  
  program_end    VARCHAR2(5) DEFAULT '16:30'  
  CHECK (  
    SUBSTR(program_end,3,1) = ':'  
    AND LENGTH(program_end) = 5  
    AND TO_NUMBER(SUBSTR(program_end,1,2)) BETWEEN 0 AND 23  
    AND TO_NUMBER(SUBSTR(program_end,4,2)) BETWEEN 0 AND 59  
  ),  
  tip_ascensor   VARCHAR2(50),  
  deschis        CHAR(1) DEFAULT 'N' CHECK (deschis IN ('Y','N'))  
);
```

```
CREATE TABLE Partie (  
  id_partie      NUMBER DEFAULT  
  seq_id_partie.NEXTVAL PRIMARY KEY,  
  denumire       VARCHAR2(100) NOT NULL,  
  dificultate    VARCHAR2(50),  
  lungime_km     NUMBER,  
  desnivel_m     NUMBER,  
  deschis        CHAR(1) DEFAULT 'N' CHECK (deschis IN ('Y','N'))  
  id_ascensor    NUMBER NOT NULL,  
  FOREIGN KEY (id_ascensor) REFERENCES Ascensor(id_ascensor)  
);
```

```
CREATE TABLE Tip_Camera (  
  id_tip_camera  NUMBER DEFAULT  
  seq_id_tip_camera.NEXTVAL PRIMARY KEY,  
  denumire       VARCHAR2(50) NOT NULL,
```

```

    tarif_zi          DECIMAL(10,2) NOT NULL CHECK (tarif_zi > 0)
);

CREATE TABLE Cabana (
    id_cabana          NUMBER DEFAULT
    seq_id_cabana.NEXTVAL PRIMARY KEY,
    nume               VARCHAR2(100) NOT NULL,
    locatie            VARCHAR2(255),
    id_partie          NUMBER NOT NULL,
    nr_camere           INTEGER NOT NULL CHECK (nr_camere >= 1),
    FOREIGN KEY (id_partie) REFERENCES Partie(id_partie)
);

CREATE TABLE Camera (
    id_camera           NUMBER DEFAULT
    seq_id_camera.NEXTVAL PRIMARY KEY,
    id_cabana           NUMBER NOT NULL,
    nr_paturi           INTEGER NOT NULL,
    capacitate_pers     INTEGER,
    id_tip_camera       NUMBER NOT NULL,
    FOREIGN KEY (id_cabana) REFERENCES Cabana(id_cabana),
    FOREIGN KEY (id_tip_camera) REFERENCES Tip_Camera(id_tip_camera)
);

CREATE TABLE Tip_Calitate (
    id_tip_calitate     NUMBER DEFAULT
    seq_id_tip_calitate.NEXTVAL PRIMARY KEY,
    denumire            VARCHAR2(50) NOT NULL,
    garantie_ani        INTEGER DEFAULT 0 CHECK (garantie_ani >= 0)
);

CREATE TABLE Echipament (
    id_echipament       NUMBER DEFAULT
    seq_id_echipament.NEXTVAL PRIMARY KEY,
    id_tip_calitate     NUMBER NOT NULL,
    denumire            VARCHAR2(100) NOT NULL,
    stoc_total          INTEGER NOT NULL CHECK (stoc_total >= 0),
    stoc_disponibil     INTEGER NOT NULL CHECK (stoc_disponibil >= 0),
    FOREIGN KEY (id_tip_calitate) REFERENCES Tip_Calitate(id_tip_calitate)
);

CREATE TABLE Schior (
    id_schior           NUMBER DEFAULT
    seq_id_schior.NEXTVAL PRIMARY KEY,
    nume                VARCHAR2(100) NOT NULL,
    telefon             VARCHAR2(20),
    email               VARCHAR2(100) UNIQUE,
    nivel_experienta    VARCHAR2(50),
    data_nasterii       DATE
);

```

```

CREATE TABLE Abonament (
    id_abonament    NUMBER DEFAULT
    seq_id_abonament.NEXTVAL PRIMARY KEY,
    id_schior       NUMBER NOT NULL,
    data_emitere    DATE DEFAULT SYSDATE,
    data_expirare   DATE,
    pret            DECIMAL(10,2) NOT NULL CHECK (pret > 0),
    FOREIGN KEY (id_schior) REFERENCES Schior(id_schior)
);

CREATE TABLE Eveniment (
    id_eveniment    NUMBER DEFAULT
    seq_id_eveniment.NEXTVAL PRIMARY KEY,
    nume            VARCHAR2(100) NOT NULL UNIQUE,
    data_eveniment  DATE,
    id_cabana       NUMBER NOT NULL,
    FOREIGN KEY (id_cabana) REFERENCES Cabana(id_cabana)
    capacitate_max INTEGER CHECK (capacitate_max >= 0)
);

CREATE TABLE Bilet (
    id_bilet        NUMBER DEFAULT
    seq_id_bilet.NEXTVAL PRIMARY KEY,
    id_schior       NUMBER NOT NULL,
    id_eveniment    NUMBER NOT NULL,
    data_emitere    DATE DEFAULT SYSDATE,
    pret            DECIMAL(10,2) NOT NULL CHECK (pret > 0),
    FOREIGN KEY (id_schior) REFERENCES Schior(id_schior),
    FOREIGN KEY (id_eveniment) REFERENCES Eveniment(id_eveniment)
);

CREATE TABLE Instructor (
    id_instructor   NUMBER DEFAULT
    seq_id_instructor.NEXTVAL PRIMARY KEY,
    nume            VARCHAR2(100) NOT NULL,
    specializare    VARCHAR2(100)
);

CREATE TABLE Lectie_Schi (
    id_lectie       NUMBER DEFAULT seq_id_lectie.NEXTVAL PRIMARY KEY,
    id_instructor   NUMBER NOT NULL,
    id_schior       NUMBER NOT NULL,
    data_lectie     DATE,
    durata_ore      INTEGER CHECK (durata_ore > 0),
    FOREIGN KEY (id_instructor) REFERENCES Instructor(id_instructor),
    FOREIGN KEY (id_schior) REFERENCES Schior(id_schior)
);

CREATE TABLE Rezervare_Camera (
    id_rezervare_camera NUMBER DEFAULT

```

```

seq_id_rezervare_camera.NEXTVAL PRIMARY KEY,
id_schior          NUMBER NOT NULL,
data_sosire        DATE NOT NULL,
durata_nopti       INTEGER NOT NULL CHECK (durata_nopti > 0),
nr_camere          INTEGER NOT NULL CHECK (nr_camere > 0),
tarif_total        DECIMAL(10,2) NOT NULL CHECK (tarif_total > 0),
FOREIGN KEY (id_schior) REFERENCES Schior(id_schior)
);

CREATE TABLE Rezervare (
    id_rezervare      NUMBER DEFAULT
seq_id_rezervare.NEXTVAL PRIMARY KEY,
id_rezervare_camera NUMBER NOT NULL,
id_camera           NUMBER NOT NULL,
data_sosire         DATE NOT NULL,
durata_nopti        INTEGER NOT NULL CHECK (durata_nopti > 0),
tarif_total         DECIMAL(10,2) NOT NULL CHECK (tarif_total > 0),
FOREIGN KEY (id_rezervare_camera)
REFERENCES Rezervare_Camera(id_rezervare_camera),
FOREIGN KEY (id_camera)           REFERENCES Camera(id_camera)
);

CREATE TABLE Rezervare_Echipament (
    id_rezervare_echipament NUMBER DEFAULT
seq_id_rezervare_echipament.NEXTVAL PRIMARY KEY,
id_schior          NUMBER NOT NULL,
data_inceput       DATE NOT NULL,
data_final         DATE NOT NULL,
numar_echipamente  INTEGER NOT NULL CHECK (numar_echipamente > 0),
tarif_total        DECIMAL(10,2) NOT NULL CHECK (tarif_total > 0),
FOREIGN KEY (id_schior) REFERENCES Schior(id_schior)
);

CREATE TABLE Inchiriere (
    id_inchiriere      NUMBER DEFAULT
seq_id_inchiriere.NEXTVAL PRIMARY KEY,
id_rezervare_echipament NUMBER NOT NULL,
id_echipament         NUMBER NOT NULL,
data_inceput          DATE NOT NULL,
data_final            DATE NOT NULL,
tarif_total           DECIMAL(10,2) NOT NULL CHECK (tarif_total > 0),
observatii_optionale  VARCHAR2(255),
FOREIGN KEY (id_rezervare_echipament) REFERENCES
Rezervare_Echipament(id_rezervare_echipament),
FOREIGN KEY (id_echipament)           REFERENCES Echipament(id_echipament)
);

CREATE TABLE Angajat (
    id_angajat NUMBER DEFAULT seq_id_angajat.NEXTVAL PRIMARY KEY,
    nume       VARCHAR2(100) NOT NULL,

```



```

    tip          VARCHAR2(50)
);

CREATE TABLE Angajat_Cabana (
    id_angajat_cabana NUMBER DEFAULT
    seq_id_angajat_cabana.NEXTVAL PRIMARY KEY,
    id_angajat        NUMBER NOT NULL,
    id_cabana          NUMBER NOT NULL,
    FOREIGN KEY (id_angajat) REFERENCES Angajat(id_angajat),
    FOREIGN KEY (id_cabana)  REFERENCES Cabana(id_cabana)
);

CREATE TABLE Angajat_Eveniment (
    id_angajat_eveniment NUMBER DEFAULT
    seq_id_angajat_eveniment.NEXTVAL PRIMARY KEY,
    id_angajat            NUMBER NOT NULL,
    id_eveniment          NUMBER NOT NULL,
    FOREIGN KEY (id_angajat) REFERENCES Angajat(id_angajat),
    FOREIGN KEY (id_eveniment) REFERENCES Eveniment(id_eveniment)
);

CREATE TABLE Angajat_Ascensor (
    id_angajat_ascensor NUMBER DEFAULT
    seq_id_angajat_ascensor.NEXTVAL PRIMARY KEY,
    id_angajat          NUMBER NOT NULL,
    id_ascensor          NUMBER NOT NULL,
    FOREIGN KEY (id_angajat) REFERENCES Angajat(id_angajat),
    FOREIGN KEY (id_ascensor) REFERENCES Ascensor(id_ascensor)
);

CREATE TABLE Intrare_Ascensor (
    id_intrare          NUMBER DEFAULT
    seq_id_intrare.NEXTVAL PRIMARY KEY,
    id_abonament        NUMBER NOT NULL,
    id_ascensor          NUMBER NOT NULL,
    data_intrare        DATE NOT NULL,
    FOREIGN KEY (id_abonament) REFERENCES Abonament(id_abonament),
    FOREIGN KEY (id_ascensor) REFERENCES Ascensor(id_ascensor)
);

```

11.2 Inserarea datelor in tabele

Creearea tabelelor SQL

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Single',          120);

```

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Double',          200);

```

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Triple',          250);

```

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Apartament',      350);

```

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Family',          400);

```

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Studio',          180);

```

```

INSERT INTO Tip_Camera
    (denumire,          tarif_zi)
VALUES
    ('Deluxe',          450);

```

-----Tip Camera -----

```

INSERT INTO Tip_Calitate
    (denumire,          garantie_ani)
VALUES
    ('standard',        1);

```

```

INSERT INTO Tip_Calitate
    (denumire,          garantie_ani)
VALUES
    ('superior',        2);

```

```

INSERT INTO Tip_Calitate
    (denumire,          garantie_ani)
VALUES
    ('premium',         2);

```

```

INSERT INTO Tip_Calitate
    (denumire,          garantie_ani)
VALUES

```

```

        ('copii',          0);

INSERT INTO Tip_Calitate
    (denumire,          garantie_ani)
VALUES
    ('avansat',          3);

INSERT INTO Tip_Calitate
    (denumire,          garantie_ani)
VALUES
    ('junior',          1);

-----Tip Calitate -----

INSERT INTO Ascensor
    (denumire,          program_start, program_end,
     tip_ascensor,      deschis)
VALUES
    ('Ascensor Nord',   '07:30',          '16:30',
     'scaun',           'Y');

INSERT INTO Ascensor
    (denumire,          program_start, program_end,
     tip_ascensor,      deschis)
VALUES
    ('Ascensor Sud',    '07:30',          '16:30',
     'gondola',         'Y');

INSERT INTO Ascensor
    (denumire,          program_start, program_end,
     tip_ascensor,      deschis)
VALUES
    ('Teleschi Valea',  '07:30',          '16:30',
     'teleschi',        'N');

INSERT INTO Ascensor
    (denumire,          program_start, program_end,
     tip_ascensor,      deschis)
VALUES
    ('Ascensor Central', '07:30',          '16:30',
     'scaun',           'Y');

INSERT INTO Ascensor
    (denumire,          program_start, program_end,
     tip_ascensor,      deschis)
VALUES
    ('Gondola Junior',  '07:30',          '16:30',
     'gondola',         'Y');

```

```

INSERT INTO Ascensor
    (denumire,          program_start, program_end,
     tip_ascensor,      deschis)
VALUES
    ('Funicular Panorama', '07:30',      '16:30',
     'funicular',        'Y');

----- Ascensor -----

INSERT INTO Cabana
    (nume,          locatie,          id_partie, nr_camere)
VALUES
    ('Cabana Alpin',    'Valea Prahovei',    1,          3);

INSERT INTO Cabana
    (nume,          locatie,          id_partie, nr_camere)
VALUES
    ('Cabana Nordica',  'Valea Alba',          2,          4);

INSERT INTO Cabana
    (nume,          locatie,          id_partie, nr_camere)
VALUES
    ('Cabana Junior',   'Valea Copiilor',    5,          5);

INSERT INTO Cabana
    (nume,          locatie,          id_partie, nr_camere)
VALUES
    ('Cabana Lupului',  'Valea Lupului',    4,          3);

INSERT INTO Cabana
    (nume,          locatie,          id_partie, nr_camere)
VALUES
    ('Cabana Relax',    'Valea Verde',       3,          3);

INSERT INTO Cabana
    (nume,          locatie,          id_partie, nr_camere)
VALUES
    ('Cabana Panorama', 'Valea Panorama',    6,          6);

----- Cabana -----

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (1,          2,          2,          1);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)

```

```

VALUES
    (1,          3,          3,          3);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (1,          4,          4,          4);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (2,          2,          2,          2);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (2,          2,          2,          1);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (2,          3,          3,          3);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (2,          4,          4,          4);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (3,          1,          1,          1);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (3,          2,          2,          2);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (3,          3,          3,          3);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (3,          4,          4,          4);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)

```

```

VALUES
    (3,          4,          4,          5);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (4,          3,          3,          2);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (4,          4,          4,          4);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (4,          2,          2,          1);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (5,          2,          2,          2);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (5,          3,          3,          3);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (5,          4,          4,          5);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (6,          1,          1,          1);

INSERT INTO Camera
    (id_cabana, nr_paturi, capacitate_pers, id_tip_camera)
VALUES
    (6,          2,          2,          2);

----- Camera -----

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (1,          'Schiuri standard', 10,          7);

```

```

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (2,          'Clăpari superior', 8,          5);

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (3,          'Cască premium',    5,          4);

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (4,          'Echipament copii', 12,          11);

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (5,          'Bețe avansate',    7,          6);

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (6,          'Schiuri junior',    5,          4);

INSERT INTO Echipament
    (id_tip_calitate, denumire,          stoc_total, stoc_disponibil)
VALUES
    (1,          'Clăpari standard', 6,          4);

----- Echipament -----

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)
VALUES
    ('Popescu Ion',    '0712345678', 'ion@exemplu.com',
     'incepator',      DATE '2001-02-15');

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)
VALUES
    ('Ionescu Maria', '0722233344', 'maria@exemplu.com',
     'mediu',          DATE '1995-06-10');

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)

```

```

VALUES
    ('Vasilescu Andrei','0733123123', 'andrei@exemplu.com',
     'avansat',          DATE '1988-11-05');

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)
VALUES
    ('Dragomir Paul',    '0744556677', 'paul@exemplu.com',
     'incepator',       DATE '2002-03-20');

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)
VALUES
    ('Stan Ana',        '0755667788', 'ana@exemplu.com',
     'mediu',          DATE '1999-09-12');

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)
VALUES
    ('Matei Georgiana', '0721345678',
     'georgiana@exemplu.com','incepator',    DATE '2004-12-24');

INSERT INTO Schior
    (nume,          telefon,          email,
     nivel_experienta, data_nasterii)
VALUES
    ('Baciu Rares',     '0764123586', 'rares@exemplu.com',
     'avansat',         DATE '1997-05-07');

----- Schior -----

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (1,          DATE '2025-01-10', DATE '2025-01-20', 500);

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (2,          DATE '2025-01-15', DATE '2025-01-18', 200);

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (3,          DATE '2025-02-01', DATE '2025-02-28', 900);

```



```

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (4,          DATE '2025-01-20', DATE '2025-01-25', 300);

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (5,          DATE '2025-01-22', DATE '2025-02-01', 350);

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (6,          DATE '2025-01-28', DATE '2025-02-05', 250);

INSERT INTO Abonament
    (id_schior, data_emitere,    data_expirare,    pret)
VALUES
    (7,          DATE '2025-02-10', DATE '2025-02-18', 400);

----- Abonament -----

INSERT INTO Eveniment
    (nume,          data_eveniment,    id_cabana,    capacitate_max)
VALUES
    ('Concurs copii',    DATE '2025-02-05', 1,          40);

INSERT INTO Eveniment
    (nume,          data_eveniment,    id_cabana,    capacitate_max)
VALUES
    ('Apres-ski party',    DATE '2025-01-25', 2,          60);

INSERT INTO Eveniment
    (nume,          data_eveniment,    id_cabana,    capacitate_max)
VALUES
    ('Seara gourmet',    DATE '2025-01-30', 3,          30);

INSERT INTO Eveniment
    (nume,          data_eveniment,    id_cabana,    capacitate_max)
VALUES
    ('Lectii gratuite',    DATE '2025-02-10', 4,          25);

INSERT INTO Eveniment
    (nume,          data_eveniment,    id_cabana,    capacitate_max)
VALUES
    ('Concurs avansati',    DATE '2025-02-15', 5,          20);

INSERT INTO Eveniment
    (nume,          data_eveniment,    id_cabana,    capacitate_max)
VALUES

```

```

('Snowboard demo',    DATE '2025-02-18', 6,          50);

----- Eveniment -----

INSERT INTO Instructor
    (nume,              specializare)
VALUES
    ('Gheorghe Mihai',  'freestyle');

INSERT INTO Instructor
    (nume,              specializare)
VALUES
    ('Dumitru Elena',   'alpin');

INSERT INTO Instructor
    (nume,              specializare)
VALUES
    ('Iacob Radu',      'snowboard');

INSERT INTO Instructor
    (nume,              specializare)
VALUES
    ('Sanda Mirela',    'copii');

INSERT INTO Instructor
    (nume,              specializare)
VALUES
    ('Popa Vlad',       'trasee dificile');

INSERT INTO Instructor
    (nume,              specializare)
VALUES
    ('Alexandru Luca',  'incepatori');

----- Instructor -----

INSERT INTO Rezervare_Camera
    (id_schior, data_sosire,    durata_nopti, nr_camere, tarif_total)
VALUES
    (1,          DATE '2025-01-12', 3,          2,          600);

INSERT INTO Rezervare_Camera
    (id_schior, data_sosire,    durata_nopti, nr_camere, tarif_total)
VALUES
    (2,          DATE '2025-01-13', 4,          1,          800);

INSERT INTO Rezervare_Camera
    (id_schior, data_sosire,    durata_nopti, nr_camere, tarif_total)
VALUES

```

```

        (3,          DATE '2025-01-15', 2,          3,          1200);

INSERT INTO Rezervare_Camera
        (id_schior, data_sosire,    durata_nopti, nr_camere, tarif_total)
VALUES
        (4,          DATE '2025-01-20', 5,          1,          1250);

INSERT INTO Rezervare_Camera
        (id_schior, data_sosire,    durata_nopti, nr_camere, tarif_total)
VALUES
        (5,          DATE '2025-01-22', 2,          2,          900);

INSERT INTO Rezervare_Camera
        (id_schior, data_sosire,    durata_nopti, nr_camere, tarif_total)
VALUES
        (6,          DATE '2025-02-18', 2,          1,          300);

----- Rezervare_Camera -----

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (1,          1,          DATE '2025-01-12', 3,          300);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (1,          2,          DATE '2025-01-12', 3,          300);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (2,          3,          DATE '2025-01-13', 4,          800);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (3,          4,          DATE '2025-01-15', 2,          400);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (3,          5,          DATE '2025-01-15', 2,          400);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (3,          6,          DATE '2025-01-15', 2,          400);

INSERT INTO Rezervare

```

```

        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (4,                                7,            DATE '2025-01-20', 5,            1250);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (5,                                8,            DATE '2025-01-22', 2,            450);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (5,                                9,            DATE '2025-01-22', 2,            450);

INSERT INTO Rezervare
        (id_rezervare_camera, id_camera, data_sosire,    durata_nopti, tarif_total)
VALUES
        (6,                                10,           DATE '2025-02-18', 2,            300);

----- Rezervare -----

INSERT INTO Rezervare_Echipament (id_schior, data_inceput,
data_final, numar_echipamente, tarif_total) VALUES (1, DATE
'2025-01-12', DATE '2025-01-14', 2, 200);

INSERT INTO Rezervare_Echipament (id_schior, data_inceput,
data_final, numar_echipamente, tarif_total) VALUES (2, DATE
'2025-01-13', DATE '2025-01-15', 1, 150);

INSERT INTO Rezervare_Echipament (id_schior, data_inceput,
data_final, numar_echipamente, tarif_total) VALUES (3, DATE
'2025-01-15', DATE '2025-01-18', 3, 270);

INSERT INTO Rezervare_Echipament (id_schior, data_inceput,
data_final, numar_echipamente, tarif_total) VALUES (4, DATE
'2025-01-20', DATE '2025-01-25', 2, 330);

INSERT INTO Rezervare_Echipament (id_schior, data_inceput,
data_final, numar_echipamente, tarif_total) VALUES (5, DATE
'2025-01-22', DATE '2025-01-24', 2, 200);

INSERT INTO Rezervare_Echipament (id_schior, data_inceput,
data_final, numar_echipamente, tarif_total) VALUES (6, DATE
'2025-02-20', DATE '2025-02-22', 1, 70);

----- Rezervare_echipament -----

INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,
data_inceput, data_final, tarif_total, observatii_optionale)
VALUES (1, 1, DATE '2025-01-12', DATE '2025-01-14', 110, 'Marime

```

```
42');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (1, 7, DATE '2025-01-12', DATE '2025-01-14', 90, NULL);
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (2, 3, DATE '2025-01-13', DATE '2025-01-15', 150, 'Casca  
S');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (3, 1, DATE '2025-01-15', DATE '2025-01-18', 90, 'Marime  
44');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament  
, data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (3, 2, DATE '2025-01-15', DATE '2025-01-18', 90, NULL);
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (3, 5, DATE '2025-01-15', DATE '2025-01-18', 90,  
'Avansat');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament  
, data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (4, 6, DATE '2025-01-20', DATE '2025-01-25', 160,  
'Copil');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (4, 7, DATE '2025-01-20', DATE '2025-01-25', 170, NULL);
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament  
, data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (5, 4, DATE '2025-01-22', DATE '2025-01-24', 100,  
'Copil');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (5, 3, DATE '2025-01-22', DATE '2025-01-24', 100, 'Casca  
M');
```

```
INSERT INTO Inchiriere (id_rezervare_echipament, id_echipament,  
data_inceput, data_final, tarif_total, observatii_optionale)  
VALUES (6, 2, DATE '2025-02-20', DATE '2025-02-22', 70, NULL);
```

```
----- Inchiriere -----
```

```

INSERT INTO Angajat (nume, tip) VALUES ('Enache Raluca', 'cabana');
INSERT INTO Angajat (nume, tip) VALUES ('Barbu George', 'cabana');
INSERT INTO Angajat (nume, tip) VALUES ('Ciobanu Mihai', 'cabana');
INSERT INTO Angajat (nume, tip) VALUES ('Badea Adriana', 'cabana');
INSERT INTO Angajat (nume, tip) VALUES ('Neagu Tudor', 'cabana');
INSERT INTO Angajat (nume, tip) VALUES ('Iacob Sorin', 'cabana');
INSERT INTO Angajat (nume, tip) VALUES ('Ilie Carmen', 'eveniment');
INSERT INTO Angajat (nume, tip) VALUES ('Dobre Simona', 'eveniment');
INSERT INTO Angajat (nume, tip) VALUES ('Petrescu Mihai', 'eveniment');
INSERT INTO Angajat (nume, tip) VALUES ('Popescu Mara', 'eveniment');
INSERT INTO Angajat (nume, tip) VALUES ('Andrei Vlad', 'eveniment');
INSERT INTO Angajat (nume, tip) VALUES ('Stan Cristina', 'eveniment');
INSERT INTO Angajat (nume, tip) VALUES ('Baciu Paul', 'ascensor');
INSERT INTO Angajat (nume, tip) VALUES ('Matei Elena', 'ascensor');
INSERT INTO Angajat (nume, tip) VALUES ('Vasilescu Emil', 'ascensor');
INSERT INTO Angajat (nume, tip) VALUES ('Munteanu Alex', 'ascensor');
INSERT INTO Angajat (nume, tip) VALUES ('Marin Raluca', 'ascensor');
INSERT INTO Angajat (nume, tip) VALUES ('Pop Vlad', 'ascensor');

```

----- Angajat -----

```

INSERT INTO Angajat_Eveniment (id_angajat, id_eveniment) VALUES (7, 1);
INSERT INTO Angajat_Eveniment (id_angajat, id_eveniment) VALUES (8, 2);
INSERT INTO Angajat_Eveniment (id_angajat, id_eveniment) VALUES (9, 3);
INSERT INTO Angajat_Eveniment (id_angajat, id_eveniment) VALUES (10, 4);
INSERT INTO Angajat_Eveniment (id_angajat, id_eveniment) VALUES (11, 5);
INSERT INTO Angajat_Eveniment (id_angajat, id_eveniment) VALUES (12, 6);

```

----- Angajat_Eveniment -----

```

INSERT INTO Angajat_Ascensor (id_angajat, id_ascensor) VALUES (13, 1);
INSERT INTO Angajat_Ascensor (id_angajat, id_ascensor) VALUES (14, 2);
INSERT INTO Angajat_Ascensor (id_angajat, id_ascensor) VALUES (15, 3);
INSERT INTO Angajat_Ascensor (id_angajat, id_ascensor) VALUES (16, 4);
INSERT INTO Angajat_Ascensor (id_angajat, id_ascensor) VALUES (17, 5);
INSERT INTO Angajat_Ascensor (id_angajat, id_ascensor) VALUES (18, 6);

```

----- Angajat_Ascensor -----

```

INSERT INTO Angajat_Cabana (id_angajat, id_cabana) VALUES (1, 1);
INSERT INTO Angajat_Cabana (id_angajat, id_cabana) VALUES (2, 2);
INSERT INTO Angajat_Cabana (id_angajat, id_cabana) VALUES (3, 3);
INSERT INTO Angajat_Cabana (id_angajat, id_cabana) VALUES (4, 4);
INSERT INTO Angajat_Cabana (id_angajat, id_cabana) VALUES (5, 5);
INSERT INTO Angajat_Cabana (id_angajat, id_cabana) VALUES (6, 6);

```

----- Angajat_Cabana -----

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (1, 1, TO_DATE('2025-01-11 08:00','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (2, 2, TO_DATE('2025-01-16 09:00','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (3, 3, TO_DATE('2025-02-02 10:15','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (4, 4, TO_DATE('2025-01-21 11:40','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (5, 5, TO_DATE('2025-01-23 14:00','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (6, 6, TO_DATE('2025-01-29 08:30','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (7, 1, TO_DATE('2025-02-12 12:10','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (1, 2, TO_DATE('2025-01-12 13:00','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (2, 3, TO_DATE('2025-01-17 13:55','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (3, 4, TO_DATE('2025-02-04 15:20','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (4, 5, TO_DATE('2025-01-22 10:25','YYYY-MM-
DD HH24:MI'));
```

```
INSERT INTO Intrare_Ascensor (id_abonament, id_ascensor,
data_intrare) VALUES (5, 6, TO_DATE('2025-01-25 11:45','YYYY-MM-
DD HH24:MI'));
```

```
----- Intrare_Ascensor -----
```

```

INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (1, 1, DATE '2025-01-25', 100);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (2, 2, DATE '2025-01-20', 150);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (3, 3, DATE '2025-01-30', 120);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (4, 4, DATE '2025-02-05', 80);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (5, 5, DATE '2025-02-15', 90);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (6, 6, DATE '2025-02-18', 70);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (7, 1, DATE '2025-02-05', 95);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (1, 3, DATE '2025-01-30', 120);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (2, 5, DATE '2025-02-15', 90);
INSERT INTO Bilet (id_schior, id_eveniment, data_emitere, pret)
VALUES (3, 2, DATE '2025-01-20', 150);

```

----- Bilet -----

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (1, 1, TO_DATE('2025-01-12 10:00','YYYY-MM-DD
HH24:MI'), 2);

```

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (2, 2, TO_DATE('2025-01-13 11:00','YYYY-MM-DD
HH24:MI'), 2);

```

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (3, 3, TO_DATE('2025-01-15 12:00','YYYY-MM-DD
HH24:MI'), 1);

```

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (4, 4, TO_DATE('2025-01-20 13:00','YYYY-MM-DD
HH24:MI'), 2);

```

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (5, 5, TO_DATE('2025-01-22 14:00','YYYY-MM-DD
HH24:MI'), 1);

```

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (6, 6, TO_DATE('2025-01-28 09:00','YYYY-MM-DD
HH24:MI'), 2);

```

```

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (1, 2, TO_DATE('2025-02-05 10:00','YYYY-MM-DD

```



```

HH24:MI'), 2);

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (2, 3, TO_DATE('2025-02-07 11:00','YYYY-MM-DD
HH24:MI'), 1);

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (3, 4, TO_DATE('2025-02-10 12:00','YYYY-MM-DD
HH24:MI'), 2);

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (4, 5, TO_DATE('2025-02-14 13:00','YYYY-MM-DD
HH24:MI'), 1);

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie
, durata_ore) VALUES (5, 7, TO_DATE('2025-02-17 15:00','YYYY-MM-
DD HH24:MI'), 1);

INSERT INTO Lectie_Schi (id_instructor, id_schior, data_lectie,
durata_ore) VALUES (6, 1, TO_DATE('2025-02-19 10:00','YYYY-MM-DD
HH24:MI'), 2);

----- Lectie_Schi -----

```

12. Cereri SQL complexe

In aceasta sectiune sunt prezentate cinci cereri SQL complexe, fiecare demonstrand elemente avansate SQL conform cerintei proiectului.

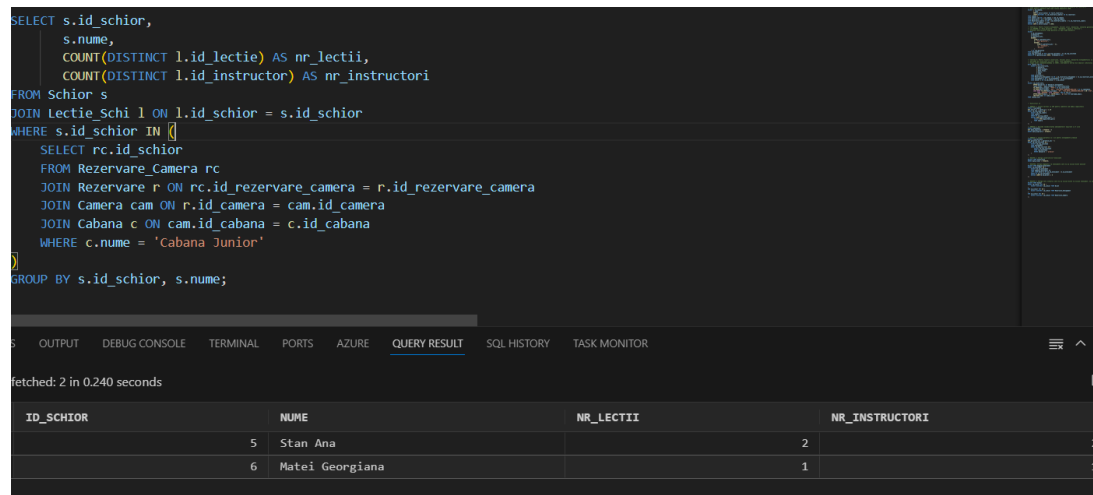
12.1 Coduri SQL pe tipuri de cerere

1. Subcerere sincronizata in SELECT (subquery corelata pe 3 tabele)

Pentru fiecare schior care a fost cazat cel putin o data la Cabana Junior, afisati id-ul, numele, numarul de lectii individuale la care a participat si numarul de instructori cu care a lucrat.

SQL

```
SELECT s.id_schior,
       s.num,
       COUNT(DISTINCT l.id_lectie) AS nr_lectii,
       COUNT(DISTINCT l.id_instructor) AS nr_instructori
FROM Schior s
JOIN Lectie_Schi l ON l.id_schior = s.id_schior
WHERE s.id_schior IN (
    SELECT rc.id_schior
    FROM Rezervare_Camera rc
    JOIN Rezervare r ON rc.id_rezervare_camera =
    r.id_rezervare_camera
    JOIN Camera cam ON r.id_camera = cam.id_camera
    JOIN Cabana c ON cam.id_cabana = c.id_cabana
    WHERE c.num = 'Cabana Junior'
)
GROUP BY s.id_schior, s.num;
```



The screenshot shows a SQL IDE interface with a dark theme. The top pane displays the SQL query from the previous block. The bottom pane shows the query results in a table format. The table has four columns: ID_SCHIOR, NUM, NR_LLECTII, and NR_INSTRUCTORI. There are two rows of data. The first row shows ID_SCHIOR 5, NUM Stan Ana, NR_LLECTII 2, and NR_INSTRUCTORI 2. The second row shows ID_SCHIOR 6, NUM Matei Georgiana, NR_LLECTII 1, and NR_INSTRUCTORI 1. The interface also includes tabs for OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, AZURE, QUERY RESULT (selected), SQL HISTORY, and TASK MONITOR. A status bar at the bottom indicates 'fetched: 2 in 0.240 seconds'.

ID_SCHIOR	NUM	NR_LLECTII	NR_INSTRUCTORI
5	Stan Ana	2	2
6	Matei Georgiana	1	1

Figura 3: Cererea 1

2. Subcerere nesincronizata in FROM (derived table)

Pentru fiecare tip de camera, afisati denumirea, tariful, numarul total de camere si media capacitatii de persoane.

SQL

```
SELECT t.denumire,
       t.tarif_zi,
       stats.nr_camere,
       stats.capacitate_medie
FROM Tip_Camera t
JOIN (
    SELECT id_tip_camera,
           COUNT(*) AS nr_camere,
           ROUND(AVG(capacitate_pers),2) AS capacitate_medie
    FROM Camera
    GROUP BY id_tip_camera
) stats
ON t.id_tip_camera = stats.id_tip_camera;
```

fetchd: 5 in 0.260 seconds

DENUMIRE	TARIF_ZI	NR_CAMERE	CAPACITATE_MEDIE
Single	120	5	1.4
Triple	250	4	3
Apartament	350	4	4
Double	200	5	2.2
Family	400	2	4

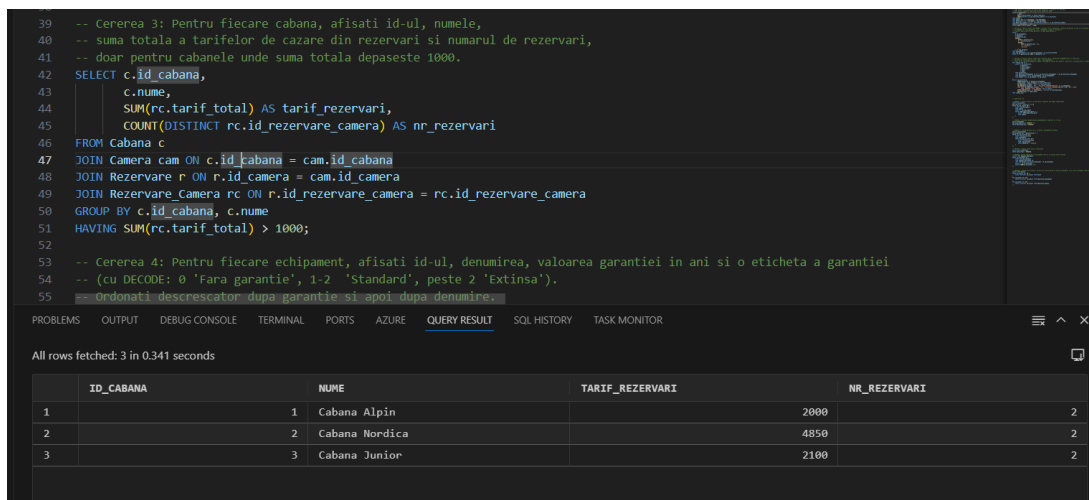
Figura 4: Cererea 2

3. Grupare si filtrare pe grupuri cu HAVING si subquery

Pentru fiecare cabana, afisati id-ul, numele, suma totala a tarifelor de cazare din rezervari si numarul de rezervari, doar pentru cabanele unde suma totala depaseste 1000.

SQL

```
SELECT c.id_cabana,
       c.num,
       SUM(rc.tarif_total) AS tarif_rezervari,
       COUNT(DISTINCT rc.id_rezervare_camera) AS nr_rezervari
FROM Cabana c
JOIN Camera cam ON c.id_cabana = cam.id_cabana
JOIN Rezervare r ON r.id_camera = cam.id_camera
JOIN Rezervare_Camera rc ON r.id_rezervare_camera =
rc.id_rezervare_camera
GROUP BY c.id_cabana, c.num
HAVING SUM(rc.tarif_total) > 1000;
```



```
39 -- Cererea 3: Pentru fiecare cabana, afisati id-ul, numele,
40 -- suma totala a tarifelor de cazare din rezervari si numarul de rezervari,
41 -- doar pentru cabanele unde suma totala depaseste 1000.
42 SELECT c.id_cabana,
43        c.num,
44        SUM(rc.tarif_total) AS tarif_rezervari,
45        COUNT(DISTINCT rc.id_rezervare_camera) AS nr_rezervari
46 FROM Cabana c
47 JOIN Camera cam ON c.id_cabana = cam.id_cabana
48 JOIN Rezervare r ON r.id_camera = cam.id_camera
49 JOIN Rezervare_Camera rc ON r.id_rezervare_camera = rc.id_rezervare_camera
50 GROUP BY c.id_cabana, c.num
51 HAVING SUM(rc.tarif_total) > 1000;
52
53 -- Cererea 4: Pentru fiecare echipament, afisati id-ul, denumirea, valoarea garantiei in ani si o eticheta a garantiei
54 -- (cu DECODE: 0 'Fara garantie', 1-2 'Standard', peste 2 'Extinsa').
55 -- Ordonati descrescator dupa garantie si apoi dupa denumire.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE QUERY RESULT SQL HISTORY TASK MONITOR

All rows fetched: 3 in 0.341 seconds

	ID_CABANA	NUME	TARIF_REZERVARI	NR_REZERVARI
1	1	Cabana Alpin	2000	2
2	2	Cabana Nordica	4850	2
3	3	Cabana Junior	2100	2

Figura 5: Cererea 3

4. Ordonari si functii NVL, DECODE

Pentru fiecare echipament, afisati id-ul, denumirea, valoarea garantiei in ani si o eticheta a garantiei (cu DECODE: 0 → 'Fara garantie', 1 → 'Standard', peste 2 → 'Extinsa'). Ordonati descrescator dupa garantie si apoi dupa denumire.

SQL

```
SELECT
    e.id_echipament,
    e.denumire,
    tc.garantie_ani,
    DECODE(
        SIGN(tc.garantie_ani),
        0, 'Fara garantie',
        DECODE(
            SIGN(tc.garantie_ani - 2),
            -1, 'Standard',
            'Extinsa'
        )
    ) AS tip_garantie
FROM Echipament e
JOIN Tip_Calitate tc ON e.id_tip_calitate = tc.id_tip_calitate
ORDER BY tc.garantie_ani DESC, e.denumire ASC;
```

-- Cererea 4: Pentru fiecare echipament, afisati id-ul, denumirea, valoarea garantiei in ani si o eticheta a garantiei
 -- (cu DECODE: 0 'Fara garantie', 1-2 'Standard', peste 2 'Extinsa').
 -- Ordonati descrescator dupa garantie si apoi dupa denumire.

```
SELECT
    e.id_echipament,
    e.denumire,
    tc.garantie_ani,
    DECODE(
        SIGN(tc.garantie_ani),
        0, 'Fara garantie',
        DECODE(
            SIGN(tc.garantie_ani - 2),
            -1, 'Standard',
            'Extinsa'
        )
    ) AS tip_garantie
FROM Echipament e
JOIN Tip_Calitate tc ON e.id_tip_calitate = tc.id_tip_calitate
ORDER BY tc.garantie_ani DESC, e.denumire ASC;
```

fetched: 7 in 0.212 seconds

ID_ECHIPAMENT	DENUMIRE	GARANTIE_ANI	TIP_GARANTIE
5	Bete avansate	3	Extinsa
3	Cască premium	2	Extinsa
2	Clăpări superior	2	Extinsa
7	Clăpări standard	1	Standard
6	Schiuri junior	1	Standard
1	Schiuri standard	1	Standard
4	Echipament copii	0	Fara garantie

Figura 6: Cererea 4

5. Bloc WITH, functii pe siruri, functii pe date calendaristice, CASE

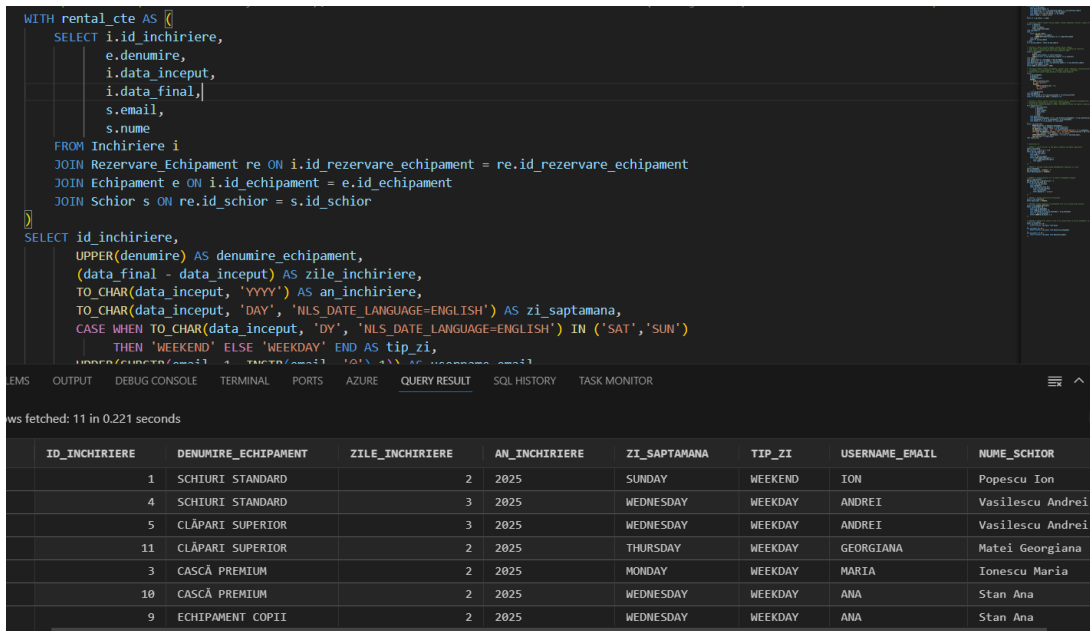
Pentru fiecare inchiriere, afisati id-ul, denumirea echipamentului cu majuscule, perioada de inchiriere in zile, anul, ziua saptamanii, tipul zilei (weekend/weekday cu CASE), username-ul extras din emailul schiorului (cu majuscule) si numele schiorului cu prima litera mare.

SQL

```

WITH rental_cte AS (
    SELECT i.id_inchiriere,
           e.denumire,
           i.data_inceput,
           i.data_final,
           s.email,
           s.numa
    FROM Inchiriere i
    JOIN Rezervare_Echipament re ON i.id_rezervare_echipament =
    re.id_rezervare_echipament
    JOIN Echipament e ON i.id_echipament = e.id_echipament
    JOIN Schior s ON re.id_schior = s.id_schior
)
SELECT id_inchiriere,
       UPPER(denumire) AS denumire_echipament,
       (data_final - data_inceput) AS zile_inchiriere,
       TO_CHAR(data_inceput, 'YYYY') AS an_inchiriere,
       TO_CHAR(data_inceput, 'DAY',
               'NLS_DATE_LANGUAGE=ENGLISH') AS
       zi_saptamana,
       CASE WHEN TO_CHAR(data_inceput, 'DY',
               'NLS_DATE_LANGUAGE=ENGLISH')
       IN ('SAT', 'SUN')
           THEN 'WEEKEND' ELSE 'WEEKDAY' END AS tip_zi,
       UPPER(SUBSTR(email, 1, INSTR(email, '@')-1))
       AS username_email,
       INITCAP(numa) AS numa_schior
FROM rental_cte;

```



The screenshot shows a SQL IDE with the query from the previous block. The 'QUERY RESULT' tab is active, displaying 11 rows of data. The table has 8 columns: ID_INCHIRIERE, DENUMIRE_ECHIPAMENT, ZILE_INCHIRIERE, AN_INCHIRIERE, ZI_SAPTAMANA, TIP_ZI, USERNAME_EMAIL, and NUMA_SCHIOR.

ID_INCHIRIERE	DENUMIRE_ECHIPAMENT	ZILE_INCHIRIERE	AN_INCHIRIERE	ZI_SAPTAMANA	TIP_ZI	USERNAME_EMAIL	NUMA_SCHIOR
1	SCHIURI STANDARD	2	2025	SUNDAY	WEEKEND	ION	Popescu Ion
4	SCHIURI STANDARD	3	2025	WEDNESDAY	WEEKDAY	ANDREI	Vasilescu Andrei
5	CLĂPARI SUPERIOR	3	2025	WEDNESDAY	WEEKDAY	ANDREI	Vasilescu Andrei
11	CLĂPARI SUPERIOR	2	2025	THURSDAY	WEEKDAY	GEORGIANA	Matei Georgiana
3	CASCĂ PREMIUM	2	2025	MONDAY	WEEKDAY	MARIA	Ionescu Maria
10	CASCĂ PREMIUM	2	2025	WEDNESDAY	WEEKDAY	ANA	Stan Ana
9	ECHIPAMENT COPII	2	2025	WEDNESDAY	WEEKDAY	ANA	Stan Ana

Figura 7: Cererea 5

13. Operații de actualizare și suprimare folosind subcereri

În această secțiune sunt prezentate trei operații de actualizare (UPDATE) și trei de ștergere (DELETE), fiecare folosind subcereri, pentru modelul bazei de date al stațiunii de schi.

13.1 Operații de actualizare (UPDATE)

1. Crește tariful cu 10% pentru camerele sub media capacității

Formulare în limbaj natural: Crește tariful cu 10% pentru toate tipurile de camere pentru care media capacității camerelor asociate este sub media capacității tuturor camerelor.

UPDATE 1

```
UPDATE Tip_Camera
SET tarif_zi = tarif_zi * 1.10
WHERE id_tip_camera IN (
    SELECT id_tip_camera
    FROM Camera
    GROUP BY id_tip_camera
    HAVING AVG(capacitate_pers) < (
        SELECT AVG(capacitate_pers)
        FROM Camera
    )
);
```

2. Crește prețul cu 15% pentru toate abonamentele deținute de schiorii care au avut cel puțin două abonamente

Formulare în limbaj natural: Crește prețul cu 15% pentru toate abonamentele schiorilor care au avut cel puțin două abonamente.

UPDATE 2

```
UPDATE Abonament
SET pret = pret * 1.15
WHERE id_schior IN (
    SELECT id_schior
    FROM Abonament
    GROUP BY id_schior
    HAVING COUNT(*) >= 2
);
```

3. Crește garanția cu 1 an pentru echipamente premium

Formulare în limbaj natural: Crește cu 1 an garanția pentru toate echipamentele de tip „premium”.

UPDATE 3

```
UPDATE Tip_Calitate
SET garantie_ani = garantie_ani + 1
WHERE id_tip_calitate IN (
    SELECT id_tip_calitate
    FROM Echipament
    WHERE id_tip_calitate IN (
        SELECT id_tip_calitate
        FROM Tip_Calitate
        WHERE denumire = 'premium'
    )
);
```

13.2 Operații de suprimare (DELETE)

1. Șterge închirierile finalizate

Formulare în limbaj natural: Șterge toate închirierile pentru care data de returnare a trecut deja.

DELETE 1

```
DELETE FROM Inchiriere
WHERE data_final < SYSDATE;
```

2. Șterge angajații la evenimente care nu au niciun bilet asociat

Formulare în limbaj natural: Șterge toate asocierile angajaților la evenimente pentru evenimente la care nu s-a emis niciun bilet.

DELETE 2

```
DELETE FROM Angajat_Eveniment
WHERE id_eveniment IN (
    SELECT ae.id_eveniment
    FROM Angajat_Eveniment ae
    LEFT JOIN Bilet b ON ae.id_eveniment = b.id_eveniment
    GROUP BY ae.id_eveniment
    HAVING COUNT(b.id_bilet) = 0
);
```

3. Șterge toți schiorii fără activitate

Formulare în limbaj natural: Șterge toți schiorii care nu au niciun bilet la niciun eveniment, nu au avut niciodată rezervare la echipament și nu au avut niciodată rezervare la cabană.

DELETE 3

```
DELETE FROM Schior
WHERE id_schior NOT IN (
    SELECT DISTINCT id_schior FROM Bilet
)
AND id_schior NOT IN (
    SELECT DISTINCT id_schior FROM Rezervare_Echipament
)
AND id_schior NOT IN (
    SELECT DISTINCT id_schior FROM Rezervare_Camera
);
```

14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.

Creem o vizualizare care să arate pentru fiecare schior:

- numele,
- emailul,
- numărul de nopți rezervate în total,
- suma totală cheltuită pe cazare.

View-ul va utiliza **JOIN** între tabelele **Schior** și **Rezervare_Camera**, precum și **GROUP BY** pentru agregare.

```
CREATE OR REPLACE VIEW v_rezervari_schiori AS
SELECT
    s.id_schior,
    s.nume,
    s.email,
    SUM(rc.durata_nopti) AS total_nopti,
    SUM(rc.tarif_total) AS total_cheltuit
FROM Schior s
JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
GROUP BY s.id_schior, s.nume, s.email;
```

Listing 1: Crearea vizualizării complexe

Exemplu de operație LMD permisă (ex: UPDATE)

Poți face UPDATE pe o vizualizare doar dacă:

- modificarea afectează exact o singură tabelă de bază,
- vizualizarea NU conține funcții de agregare sau JOIN-uri complexe.

Exemplu permis:

```
UPDATE v_contact_schior
SET email = 'noul_email@exemplu.com'
WHERE id_schior = 1;
```

Listing 2: Update permis pe vizualizare simplă

Exemplu de operație LMD nepermisă pe vizualizarea complexă

```
UPDATE v_rezervari_schiori
SET total_cheltuit = 0
WHERE id_schior = 1;
```

Listing 3: Operație nepermisă: modificare agregat

Aceste operații nu sunt permise deoarece:

- coloanele `total_nopti` și `total_cheltuit` sunt rezultate din funcții de agregare,
- view-ul folosește `GROUP BY`,
- nu există reguli clare de "inversare" a modificărilor către tabelele de bază.

15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația *outer-join* pe minimum 4 tabele, o cerere ce utilizează operația *division* și o cerere care implementează analiza *top-n*.

a) Outer Join pe minimum 4 tabele

Formulare în limbaj natural: Afișați lista tuturor schiorilor, numele instructorilor cu care au avut lecții de schi, denumirea cabanei unde au fost cazați și emailul lor. Dacă un schior nu a avut nici o lecție sau nu a fost cazat în nici o cabană, să apară totuși în rezultat.

```
SELECT
    s.num AS nume_schior,
    s.email,
    i.num AS nume_instructor,
    c.num AS nume_cabana
FROM
    Schior s
LEFT OUTER JOIN Lectie_Schi ls ON s.id_schior = ls.id_schior
LEFT OUTER JOIN Instructor i ON ls.id_instructor = i.id_instructor
LEFT OUTER JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
LEFT OUTER JOIN Rezervare r ON rc.id_rezervare_camera = r.id_rezervare_camera
LEFT OUTER JOIN Camera cam ON r.id_camera = cam.id_camera
LEFT OUTER JOIN Cabana c ON cam.id_cabana = c.id_cabana;
```

Listing 4: Cerere cu LEFT OUTER JOIN pe minim 4 tabele

b) Operația de DIVISION

Formulare în limbaj natural: Afișează schiorii care au avut rezervări la cel puțin două tipuri diferite de cameră.

```
SELECT s.num, s.email
FROM Schior s
WHERE (
```

```

SELECT COUNT(DISTINCT cam.id_tip_camera)
FROM Rezervare_Camera rc
JOIN Rezervare r ON rc.id_rezervare_camera = r.id_rezervare_camera
JOIN Camera cam ON r.id_camera = cam.id_camera
WHERE rc.id_schior = s.id_schior
) >= 2;

```

Listing 5: Cerere ce utilizează operația de DIVISION

c) Analiza TOP-N

Formulare în limbaj natural: Afișați primii 3 schiori care au cheltuit cei mai mulți bani pe rezervările de camere.

```

SELECT *
FROM (
    SELECT
        s.numa,
        s.email,
        SUM(rc.tarif_total) AS total_cheltuit
    FROM Schior s
    JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
    GROUP BY s.numa, s.email
    ORDER BY total_cheltuit DESC
)
WHERE ROWNUM <= 3;

```

Listing 6: Analiza TOP-N – primii 3 schiori după totalul cheltuit

16. Optimizarea și analiza execuției unei cereri SQL

a) Optimizarea unei cereri utilizând proprietățile algebrei relaționale

a.i. Cerere inițială

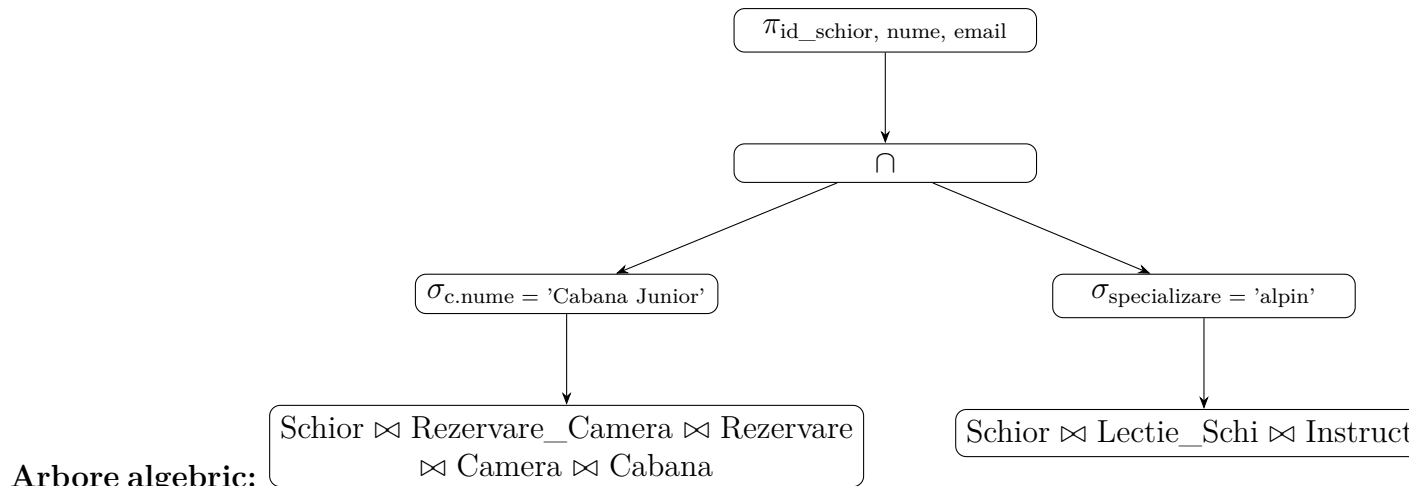
Expresie algebrică:

```

Rezultat = {_id_schior, nume, email} (
    Rezultat = {_id_schior, nume, email} (_{c.nume='Cabana Junior'}
    (Schior Rezervare_Camera Rezervare Camera Cabana))

    {_id_schior, nume, email} (Schior Lectie_Schi Instructor
    (_{specializare='alpin'}(Instructor)))

```



Arbore algebric:

SQL corespunzător:

```

SELECT DISTINCT s.nume, s.email
FROM Schior s
JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
JOIN Rezervare r ON rc.id_rezervare_camera = r.id_rezervare_camera
JOIN Camera cam ON r.id_camera = cam.id_camera
JOIN Cabana c ON cam.id_cabana = c.id_cabana
WHERE c.nume = 'Cabana Junior'
AND s.id_schior IN (
  SELECT ls.id_schior
  FROM Lectie_Schi ls
  JOIN Instructor i ON ls.id_instructor = i.id_instructor
  WHERE i.specializare = 'alpin'
);
  
```

Listing 7: Cerere inițială cu intersecție între două subcereri

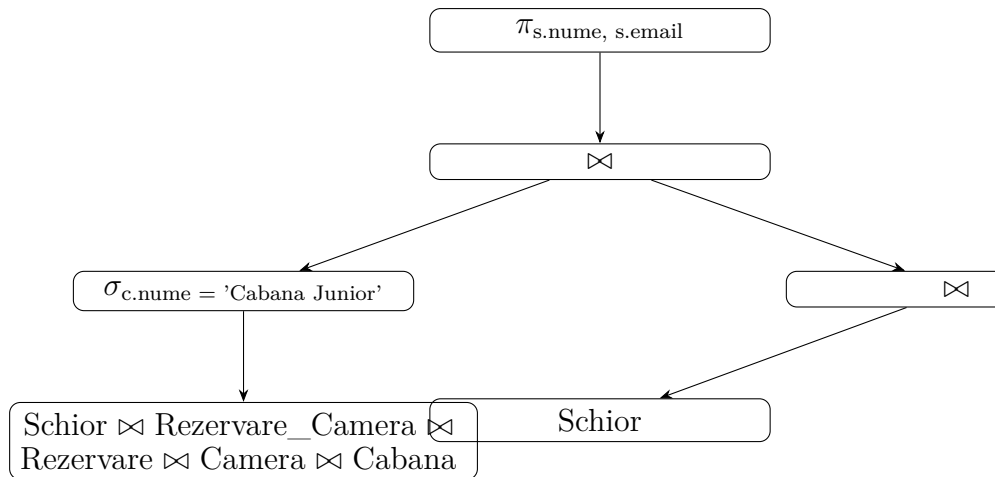
a.ii. Cerere optimizată

Expresie algebrică optimizată:

```

_{s.nume, s.email} (
  _{c.nume='Cabana Junior'}(Schior Rezervare_Camera Rezervare Camera Cabana)

  (Schior Lectie_Schi (_{specializare='alpin'}(Instructor)))
)
  
```



Arbore algebric optimizat:

SQL optimizat:

```
SELECT DISTINCT s.num, s.email
FROM Schior s
JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
JOIN Rezervare r ON rc.id_rezervare_camera = r.id_rezervare_camera
JOIN Camera cam ON r.id_camera = cam.id_camera
JOIN Cabana c ON cam.id_cabana = c.id_cabana AND c.num = 'Cabana Junior'
JOIN Lectie_Schi ls ON s.id_schior = ls.id_schior
JOIN Instructor i ON ls.id_instructor = i.id_instructor AND i.specializare = 'alpin';
```

Listing 8: Cerere optimizată cu JOIN-uri directe și filtre mutate

b) Prezentarea planului de execuție și optimizarea performanței

b.i. Cerere SQL de analiză

```
SELECT
    s.num,
    SUM(rc.tarif_total) AS total_cheltuit
FROM Schior s
JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
GROUP BY s.num
ORDER BY total_cheltuit DESC;
```

Listing 9: Sumă totală cheltuită de fiecare schior

b.ii. Plan de execuție fără optimizare

- Scanare completă (Table Scan) a tabelului Schior
- Scanare completă a tabelului Rezervare_Camera
- Join folosind Nested Loop sau Hash Join
- Grupare (Group By) după nume

- Sortare descrescătoare după `total_cheltuit`

b.iii. Optimizare cu index și hint Oracle

Creare index:

```
CREATE INDEX idx_rc_id_schior ON Rezervare_Camera(id_schior);
```

Listing 10: Crearea unui index pentru coloană de join

Cerere SQL cu hint Oracle:

```
SELECT /*+ USE_NL(rc) */
    s.numa,
    SUM(rc.tarif_total) AS total_cheltuit
FROM Schior s
JOIN Rezervare_Camera rc ON s.id_schior = rc.id_schior
GROUP BY s.numa
ORDER BY total_cheltuit DESC;
```

Listing 11: Cerere cu hint pentru Nested Loop Join

b.iv. Plan de execuție optimizat

- Index Scan pe `Rezervare_Camera` folosind `idx_rc_id_schior`
- Join rapid cu Nested Loop
- Grupare eficientă cu agregare
- Sortare descrescătoare pe câmp agregat

b.v. Comparație între execuții

Execuție fără optimizare:

- Scanare completă a tabelor
- Join lent
- Cost mare la `Group By` și `ORDER BY`

Execuție optimizată:

- Acces eficient prin index
- Join rapid
- Agregare și sortare mult mai rapide

17. Normalizare avansată și denormalizare

a. Realizarea normalizării BCNF, FN4, FN5

Exemplu non-BCNF și normalizare la BCNF

Relația analizată este:

`Rezervare(id_rezervare, id_rezervare_camera, id_camera, data_sosire, durata_nopti, tar...`

Dependențe funcționale:

$(id_rezervare, id_rezervare_camera, id_camera) \rightarrow (data_sosire, durata_nopti, tarif_total)$

$id_rezervare_camera \rightarrow (data_sosire, durata_nopti, tarif_total)$

Observație: $id_rezervare_camera$ nu este o cheie candidat, dar determină atribute nenucleu; deci relația nu este în forma normală Boyce-Codd (BCNF).

Descompunere pentru BCNF:

- Rezervare_Camera($id_rezervare_camera, data_sosire, durata_nopti, tarif_total$)
- Rezervare($id_rezervare, id_rezervare_camera, id_camera$)

Exemplu non-FN4 și normalizare la FN4

Relația analizată:

Participare($id_eveniment, id_schior, premiu$)

Pentru fiecare eveniment, pot exista mai mulți schiori participanți și mai multe premii acordate independent.

Dependențe multivaluate:

$id_eveniment \twoheadrightarrow id_schior$

$id_eveniment \twoheadrightarrow premiu$

Motiv non-FN4: Relația are două dependențe multivaluate nenule și neincluse una în cealaltă, ceea ce duce la redundanță și anomalii — deci nu este în forma normală a patra (FN4).

Descompunere pentru FN4:

- Participare1($id_eveniment, id_schior$)
- Participare2($id_eveniment, premiu$)

Exemplu non-FN5 și normalizare la FN5

Relație ipotetică:

Productie($id_fabrica, id_materie_prima, id_produs$)

Toate combinațiile de fabrică, materie primă și produs sunt posibile dacă fiecare pereche există deja în altă relație.

Motiv non-FN5: Relația permite descompunerea pe perechi, dar această descompunere poate introduce redundanță și anomalii la inserare sau ștergere. Deci relația nu este în forma normală a cincea (FN5).

Descompunere pentru FN5:

- Prod1($id_fabrica, id_materie_prima$)

- Prod2(id_fabrica, id_produs)
- Prod3(id_materie_prima, id_produs)

b. Aplicarea denormalizării și justificarea necesității

Exemplu de denormalizare și justificare

Situație inițială:

Emailurile schiorilor sunt stocate într-un tabel separat:

```
Schior(id_schior, nume, telefon, ...)
Schior_Email(id_schior, email)
```

Denormalizare (mutare email principal în tabelul Schior):

```
Schior(id_schior, nume, telefon, email, ...)
```

Justificare:

Accesul la datele de contact este realizat mai rapid, fără a necesita operații JOIN între tabele. În plus, acest model îmbunătățește performanța interogărilor frecvente (de exemplu, listarea schiorilor cu emailuri asociate). De asemenea, simplifică gestionarea datelor curente și actualizarea informațiilor de contact, fiind mai potrivit pentru operații administrative de zi cu zi.