PROIECT BAZE DE DATE

Tema proiectului - Managerierea Proiectelor

Nume: Pincu Iulia Maria Andreea
Grupa: 142

CUPRINS:

1.1. **Descrierea**

Proiectul este o aplicație de gestionare a proiectelor care implică colaborarea între clienți, firme, angajați și manageri. Scopul acestui proiect este de a oferi o platformă centralizată pentru administrarea și monitorizarea proiectelor într-un mod eficient și organizat.

Aplicația de gestionare a proiectelor facilitează comunicarea și coordonarea între diferitele părți implicate într-un proiect. Ea permite clienților să înregistreze proiectele lor, să monitorizeze stadiile și să primească actualizări în timp real. Firmele pot utiliza aplicația pentru a organiza și aloca resursele necesare proiectelor. Pentru angajați și manageri, aplicația de gestionare a proiectelor reprezintă un instrument esențial pentru administrarea sarcinilor și raportarea progresului. Angajații pot vedea sarcinile alocate, termenele limită și prioritățile acestora, având astfel o imagine clară a responsabilităților lor.

Prin centralizarea datelor și informațiilor într-o bază de date, aplicația de gestionare a proiectelor permite accesul rapid și ușor la informații actualizate despre proiecte, clienți, firme, angajați și alte detalii relevante. Astfel, se asigură transparența, eficiența și coerența în gestionarea proiectelor și se optimizează procesele de luare a deciziilor.

1.2. **Regulile de funcționare:**

- Un client poate avea mai multe contracte cu diferite firme, iar o firmă poate avea mai mulți clienți.
- Un contract este o legătură între un client și o firma care atestă colaborarea dintre aceștia
- Fiecare proiect trebuie să aibă un manager responsabil, iar un manager poate fi asociat cu mai multe proiecte.
- Un proiect trebuie să aibă o perioadă specificată, cu o dată de început și o dată de finalizare.
- Pentru fiecare proiect, există un set de task-uri care trebuie realizate de către angajați pana la deadline.
- Angajații sunt repartizați în birouri situate la anumite locații care facilitează localizarea lor cât și impărțirea acestora în funcție de proiectele la care lucrează.
- Există ședințe periodice la care participă managerii și angajații. Un angajat poate participa la mai multe ședințe, iar o ședință poate avea mai mulți participanți.

Aceste reguli asigură o structură coerentă și organizată a datelor din baza de date, precum și o gestionare eficientă a proiectelor și a resurselor implicate.

2. Prezentarea Constrângerilor impuse asupra modelului

- Un proiect poate avea un singur manager
- In cadrul unei sedinte participa obligatoriu un manager
- Un angajat ocupa un singur loc într-un birou
- Un proiect poate fi realizat de o singură firmă

3.Entități

Proiectul de gestionare a proiectelor implică utilizarea mai multor entități pentru a stoca și gestiona informațiile relevante. Voi enumera fiecare entitate în parte și voi furniza o descriere succintă a rolului său în cadrul bazei de date, precum și specificarea cheii primare asociate.

Entități:

Proiectul de gestionare a proiectelor implică utilizarea mai multor entități pentru a stoca și gestiona informațiile relevante. Voi enumera fiecare entitate în parte și voi furniza o descriere succintă a rolului său în cadrul bazei de date, precum și specificarea cheii primare asociate.

1. **CLIENT**: reprezintă informațiile despre clienții implicați în proiecte. Această entitate conține numele și un identificator unic pentru fiecare client.
    - CHEIA PRIMARĂ: ID_CLIENT
2. **FIRMA**: stochează informațiile despre firmele asociate proiectelor. Această entitate conține numele firmei, codul CAEN (Clasificarea Activităților din Economia Națională) și un identificator unic pentru fiecare firmă.
    - CHEIA PRIMARĂ: CUI_FIRMA
3. **CONTRACT**: este o entitate de legătură între clienți și firme, reflectând contractele încheiate între acestea. Aceasta reține informații despre asocierile dintre clienți și firme și servește la realizarea relațiilor de mulți-la-mulți între cele două entități.
    - CHEIA PRIMARĂ: (ID_CLIENT, CUI_FIRMA)
    - CHEI STRĂINE: ID_CLIENT referențiază tabela CLIENT(ID_CLIENT), CUI_FIRMA referențiază tabela FIRMA(CUI_FIRMA)

4. **PERIOADA**: reprezintă perioadele de timp asociate proiectelor. Această entitate stochează informații despre data de început și data de finalizare a unei perioade specifice.
    - CHEIA PRIMARĂ: ID_PERIOADA
5. **LOCATIE**: conține informații despre diferite locații sau adrese care pot fi asociate birourilor.
    - CHEIA PRIMARĂ: ID_LOCATIE
6. **BIROU**: reprezintă birourile din cadrul proiectelor și este asociat cu o anumită locație. Această entitate conține informații despre adresa biroului și capacitatea acestuia.
    - CHEIA PRIMARĂ: ID_BIROU
    - CHEIE STRĂINĂ: ID_LOCATIE referențiază tabela LOCATIE(ID_LOCATIE)
7. **ANGAJAT**: stochează informațiile despre angajații implicați în proiecte, cum ar fi nume, prenume, salariu, data de naștere și biroul în care lucrează.
    - CHEIA PRIMARĂ: ID_ANGAJAT
    - CHEIE STRĂINĂ: ID_BIROU referențiază tabela BIROU(ID_BIROU)
8. **MANAGER**: reprezintă managerii proiectelor și conține informații despre aceștia, precum nume, prenume, data de naștere și ani de experiență.
    - CHEIA PRIMARĂ: ID_MANAGER
9. **PROIECT**: reprezintă proiectele în sine și conține informații despre numele proiectului, managerul asociat, perioada și firma responsabilă pentru proiect.
    - CHEIA PRIMARĂ: ID_PROIECT
    - CHEI STRĂINE: ID_MANAGER referențiază tabela MANAGER(ID_MANAGER), ID_PERIOADA referențiază tabela PERIOADA(ID_PERIOADA), CUI_FIRMA referențiază tabela FIRMA(CUI_FIRMA)
10. **TASK**: stochează informații despre sarcinile sau task-urile atribuite angajaților în cadrul unui proiect. Această entitate conține detalii precum deadline-ul, dificultatea, durata estimată și asocierea cu un anumit angajat și proiect.
    - CHEIA PRIMARĂ: ID_TASK
    - CHEI STRĂINE: ID_ANGAJAT referențiază tabela ANGAJAT(ID_ANGAJAT), ID_PROIECT referențiază tabela PROIECT(ID_PROIECT)

11. **SEDINTA**: reprezintă ședințele managerilor și conține informații despre ora de început, ora de sfârșit, managerul responsabil și data ședinței.
    - CHEIA PRIMARĂ: ID_SEDINTA
    - CHEIE STRĂINĂ: ID_MANAGER referențiază tabela MANAGER(ID_MANAGER)
12. **PARTICIPARE_SEDINTA**: stochează informații despre participarea angajaților la ședințe, cum ar fi confirmarea și legătura cu ședința și angajatul.
    - CHEIE STRĂINĂ: ID_SEDINTA referențiază tabela SEDINTA(ID_SEDINTA), ID_ANGAJAT referențiază tabela ANGAJAT(ID_ANGAJAT)

Aceasta este structura generală a entităților și cheilor primare asociate în cadrul bazei de date.

4.Descrierea relatiilor incluzand si cardinalitatile

Relațiile dintre entități:

CLIENT_semnează_CONTRACT = CLIENT (1) ----- (1 sau mai multe) CONTRACT

Un client poate semna unul sau mai multe contracte asociate. Un contract este asociat unui singur client.

FIRMA_semnează_CONTRACT = FIRMA (1) ----- (1 sau mai multe) CONTRACT

O firmă poate avea unul sau mai multe contracte asociate. Un contract este asociat unei singure firme.

Relația între entitățile CLIENT și FIRMA prin intermediul entității CONTRACT = CLIENT (1 sau mai multe) ----- (1 sau mai multe) CONTRACT ----- (1 sau mai multe) FIRMA

Un client poate avea unul sau mai multe contracte și o firmă poate avea unul sau mai multe contracte. Această relație permite asocierea între clienți și firme prin intermediul contractelor.

PERIOADA_se_desfășoară_PROIECT = PERIOADA (1) ----- (0 sau mai multe) PROIECT

Unui proiect ii este asociată o singură perioada, dar mai multe proiecte se pot desfășura în aceeași perioada.

BIROU_are_LOCATIE = LOCATIE (1) ----- (1 sau mai multe) BIROU

O locație poate avea unul sau mai multe birouri, iar un birou este asociat unei singure locații.

ANGAJAT_are_BIROU = BIROU (1) ----- (1 sau mai multe) ANGAJAT

Un birou poate avea zero sau mai mulți angajați, iar un angajat poate fi asociat unui singur birou.

MANAGER_gestionează_PROIECT = MANAGER (1) ----- (0 sau mai multe) PROIECT

Un manager poate fi asociat mai multor proiecte, iar un proiect are asociat un singur manager care il gestionează.

ANGAJAT_are_TASK = ANGAJAT (1) ----- (1 sau mai multe) TASK

Un angajat poate avea una sau mai multe sarcini atribuite, iar o sarcină poate fi atribuită unui singur angajat.

MANAGER_participă_SEDINTA = MANAGER (1) ----- (0 sau mai multe) SEDINTA

Un manager poate organiza zero sau mai multe ședințe, iar o ședință poate fi organizată de un singur manager.

Relația între entitățile ANGAJAT și SEDINTA prin intermediul entității PARTICIPARE_SEDINTA = ANGAJAT (1 sau mai multe) ----- (0 sau mai multe) PARTICIPARE_SEDINTA ----- (0 sau mai multe) SEDINTA

Un angajat poate participa la zero sau mai multe ședințe, iar o ședință poate avea unul sau mai mulți angajați participanți.

5.Descrierea atributelor

ENTITATE: CLIENT

- *ID_CLIENT*:
  - reprezintă identificatorul unic al clientului.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY (cheie primară), NOT NULL
- *NUME*:
  - reprezintă numele clientului.
  - Tip de date: VARCHAR2(20)
  - Constrângeri: NOT NULL

ENTITATE: FIRMA

- *CUI_FIRMA*:
  - reprezintă Codul Unic de Identificare al firmei.
  - Tip de date: NUMBER(10)
  - Constrângeri: PRIMARY KEY, NOT NULL
- *NUME_FIRMA*:
  - reprezintă numele firmei.
  - Tip de date: VARCHAR2(20)
  - Constrângeri: NOT NULL

- *COD_CAEN*:
  - reprezintă Codul CAEN (Clasificarea Activităților din Economia Națională) al firmei.
  - Tip de date: NUMBER(10)
  - Constrângeri: NOT NULL

ENTITATE: CONTRACT

- *ID_CLIENT*:
  - reprezintă identificatorul clientului asociat contractului.
  - Tip de date: NUMBER(4)
  - Constrângeri: FOREIGN KEY către CLIENT(ID_CLIENT)
- *CUI_FIRMA*:
  - reprezintă Codul Unic de Identificare al firmei asociate contractului.
  - Tip de date: NUMBER(10)
  - Constrângeri: FOREIGN KEY către FIRMA(CUI_FIRMA)
  - Cheie primară compusă din ID_CLIENT și CUI_FIRMA.

ENTITATE: PERIOADA

- *ID_PERIOADA*:
  - reprezintă identificatorul unic al perioadei.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY
- *DATA_START*:
  - reprezintă data de început a perioadei.
  - Tip de date: DATE
  - Constrângeri: DEFAULT SYSDATE (valoarea implicită este data curentă)
- *DATA_FINISH*:
  - reprezintă data de încheiere a perioadei.
  - Tip de date: DATE
  - Constrângeri: DEFAULT SYSDATE (valoarea implicită este data curentă)

ENTITATE: LOCATIE

- *ID_LOCATIE*:
  - reprezintă identificatorul unic al locației.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY
- *ADRESA*:
  - reprezintă adresa locației.
  - Tip de date: VARCHAR2(20)
  - Constrângeri: NOT NULL

ENTITATE: BIROU

- *ID_BIROU*:
  - reprezintă identificatorul unic al biroului.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY
- *ID_LOCATIE*:
  - reprezintă identificatorul locației asociate biroului.
  - Tip de date: NUMBER(4)
  - Constrângeri: FOREIGN KEY către LOCATIE(ID_LOCATIE)
- *ADRESA*:
  - reprezintă adresa biroului.
  - Tip de date: VARCHAR2(50)
  - Constrângeri: NOT NULL
- *CAPACITATE*:
  - reprezintă capacitatea biroului (numărul maxim de angajați care pot lucra în birou).
  - Tip de date: NUMBER(1)

ENTITATE: ANGAJAT

- *ID_ANGAJAT*:
  - reprezintă identificatorul unic al angajatului.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY
- *NUME*:
  - reprezintă numele angajatului.
  - Tip de date: VARCHAR2(10)
- *PRENUME*:
  - reprezintă prenumele angajatului.
  - Tip de date: VARCHAR2(10)
- *SALARIU*:
  - reprezintă salariul angajatului.
  - Tip de date: NUMBER(6,2)
- *DATA_NASTERII*:
  - reprezintă data nașterii angajatului.
  - Tip de date: DATE
- *ID_BIROU*:
  - reprezintă identificatorul biroului în care lucrează angajatul.
  - Tip de date: NUMBER(4)
  - Constrângeri: FOREIGN KEY către BIROU(ID_BIROU)

ENTITATE: MANAGER

- *ID_MANAGER*:
  - reprezintă identificatorul unic al managerului.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY

- *NUME*:
  - reprezintă numele managerului.
  - Tip de date: VARCHAR2(20)
- *PRENUME*:
  - reprezintă prenumele managerului.
  - Tip de date: VARCHAR2(20)
-
- *DATA_NASTERII*:
  - reprezintă data nașterii managerului.
  - Tip de date: DATE
- *ANI_EXPERIENTA*:
  - reprezintă numărul de ani de experiență al managerului.
  - Tip de date: NUMBER(4)

ENTITATE: PROIECT

- *ID_PROIECT*:
  - reprezintă identificatorul unic al proiectului.
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY
- *NUME_PROIECT*:
  - reprezintă numele proiectului.
  - Tip de date: VARCHAR2(15)
- *ID_MANAGER*:
  - reprezintă identificatorul managerului responsabil de proiect.
  - Tip de date: NUMBER(4)
  - Constrângeri: FOREIGN KEY către MANAGER(ID_MANAGER)
- *ID_PERIOADA*:
  - reprezintă identificatorul perioadei în care se desfășoară proiectul.
  - Tip de date: NUMBER(4)
  - Constrângeri: FOREIGN KEY către PERIOADA(ID_PERIOADA)
- *CUI_FIRMA*:
  - reprezintă Codul Unic de Identificare al firmei responsabile de proiect.
  - Tip de date: NUMBER(10)
  - Constrângeri: FOREIGN KEY către FIRMA(CUI_FIRMA)

ENTITATE: TASK

- *ID_TASK*:
  - reprezintă identificatorul unic al sarcinii (task-ului).
  - Tip de date: NUMBER(4)
  - Constrângeri: PRIMARY KEY
- *DEADLINE*:
  - reprezintă data limită pentru finalizarea sarcinii.
  - Tip de date: DATE
- *DIFICULTATE*:

- ➢ reprezintă nivelul de dificultate al sarcinii.
- ➢ Tip de date: VARCHAR2(5)

❖ *DURATA_ORE*:
- ➢ reprezintă durata estimată în ore pentru finalizarea sarcinii.
- ➢ Tip de date: NUMBER(4,1)

❖

❖ *ID_ANGAJAT*:
- ➢ reprezintă identificatorul angajatului responsabil de sarcină.
- ➢ Tip de date: NUMBER(4)
- ➢ Constrângeri: FOREIGN KEY către ANGAJAT(ID_ANGAJAT)

❖ *ID_PROIECT*:
- ➢ reprezintă identificatorul proiectului căruia îi este asignată sarcina.
- ➢ Tip de date: NUMBER(4)
- ➢ Constrângeri: FOREIGN KEY către PROIECT(ID_PROIECT)

ENTITATE: SEDINTA

❖ *ID_SEDINTA*:
- ➢ reprezintă identificatorul unic al ședinței.
- ➢ Tip de date: NUMBER(4)
- ➢ Constrângeri: PRIMARY KEY

❖ *ORA_START*:
- ➢ reprezintă ora de începere a ședinței.
- ➢ Tip de date: VARCHAR2(8)

❖ *ORA_FINISH*:
- ➢ reprezintă ora de încheiere a ședinței.
- ➢ Tip de date: VARCHAR2(8)

❖ *ID_MANAGER*:
- ➢ reprezintă identificatorul managerului care organizează ședința.
- ➢ Tip de date: NUMBER(4)
- ➢ Constrângeri: FOREIGN KEY către MANAGER(ID_MANAGER)

❖ *DATA*:
- ➢ reprezintă data la care are loc ședința.
- ➢ Tip de date: DATE

ENTITATE: PARTICIPARE_SEDINTA

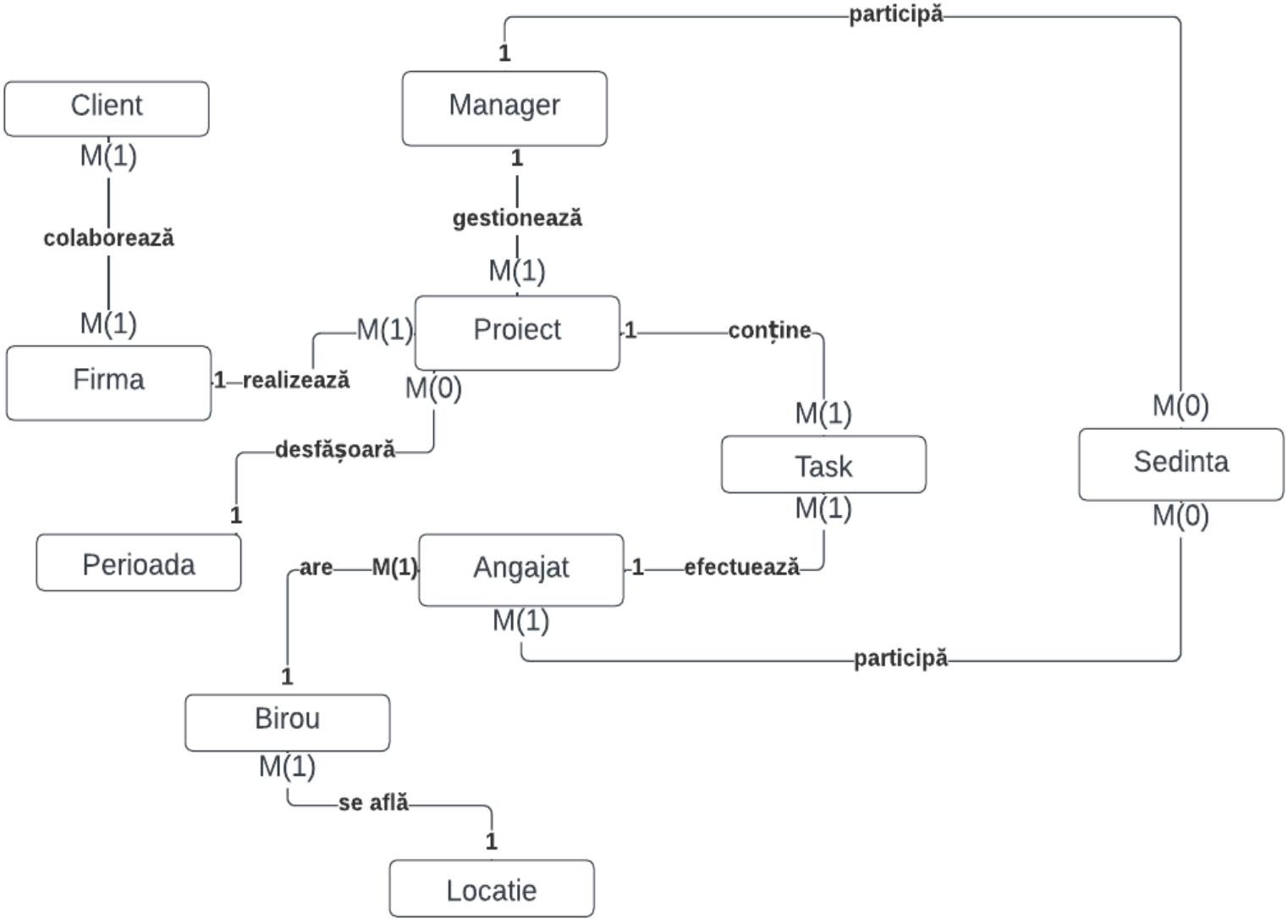
❖ *ID_SEDINTA*:
- ➢ reprezintă identificatorul ședinței la care angajatul participă.
- ➢ Tip de date: NUMBER(4)
- ➢ Constrângeri: FOREIGN KEY către SEDINTA(ID_SEDINTA)

❖ *ID_ANGAJAT*:
- ➢ reprezintă identificatorul angajatului care participă la ședință.
- ➢ Tip de date: NUMBER(4)
- ➢ Constrângeri: FOREIGN KEY către ANGAJAT(ID_ANGAJAT)

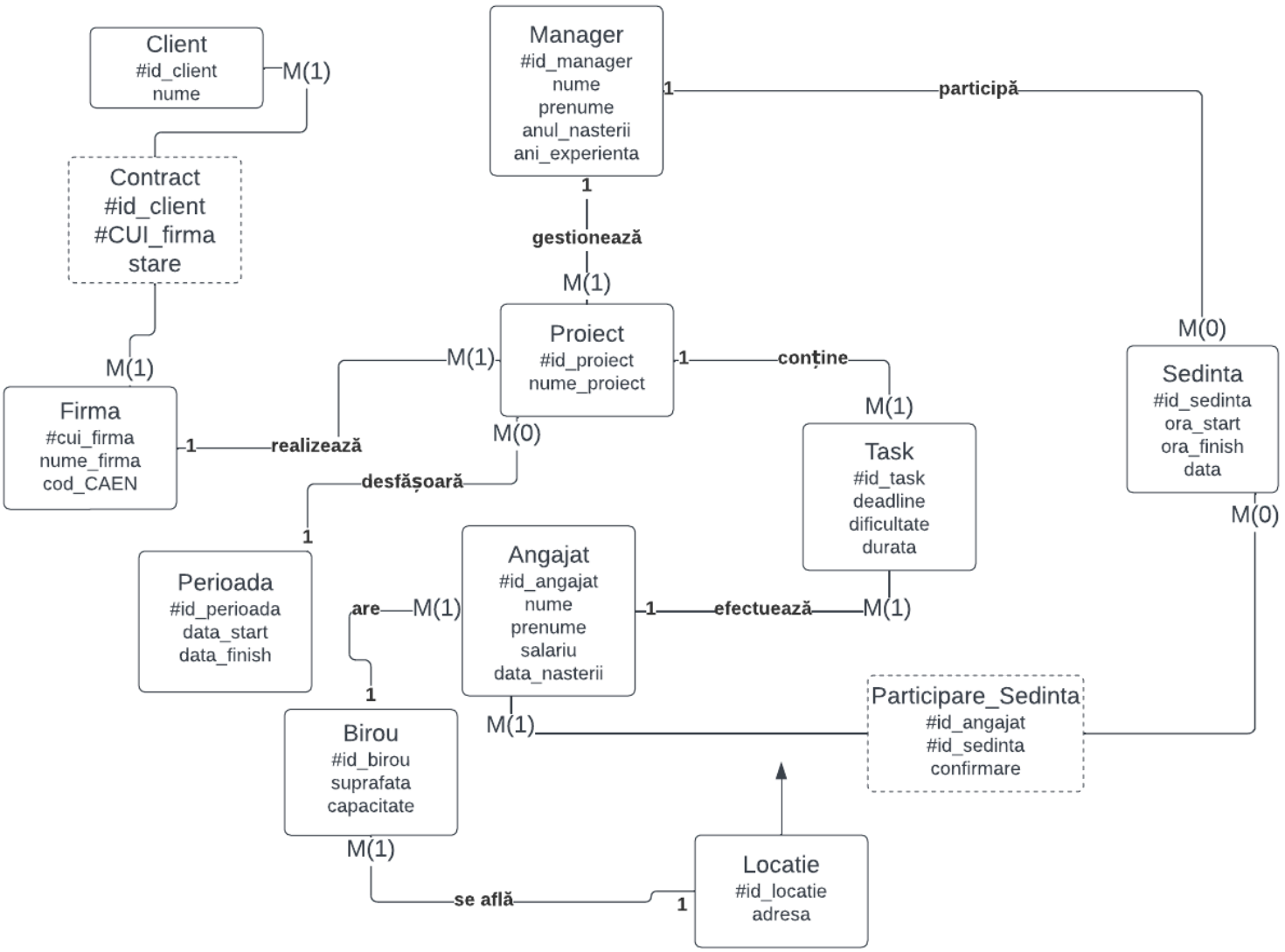
❖ *CONFIRMARE*:

- reprezintă confirmarea (confirmat/neconfirmat) angajatului la ședință.
- Tip de date: VARCHAR(20)

6.ERD



7.Diagrama Conceptuala



8.Schemele relaționale

CLIENT (#id_client, nume)

FIRMA (#cui_firma, nume_firma, cod_caen)

CONTRACT (id_client, cui_firma)

PERIOADA (#id_perioada, data_start, data_finish)

LOCATIE (#id_locatie, adresa)

BIROU (#id_birou, id_locatie, adresa, capacitate)

ANGAJAT (#id_angajat, nume, prenume, salariu, data_nasterii, id_birou)

MANAGER (#id_manager, nume, prenume, data_nasterii, ani_experienta)

PROIECT (#id_proiect, nume_proiect, id_manager, id_perioada, cui_firma)

TASK (#id_task, deadline, dificultate, durata_ore, id_angajat, id_proiect)

SEDINTA (#id_sedinta, ora_start, ora_finish, id_manager, data)

PARTICIPARE_SEDINTA (id_sedinta, id_angajat, confirmare)

9.FN1-FN3

★ FN1

O relație se află în Forma Normală 1 (FN1) atunci când nu există duplicări de date într-un singur rând și toate atributele conțin doar valori atomice (necomponate). În FN1, o relație este organizată într-un singur tabel si are un identificator unic.

Exemple care dovedesc faptul ca diagrama se află în FN1:

- Tabelul "CLIENT" este în FN1 deoarece are o cheie primară (ID_CLIENT) care identifică în mod unic fiecare înregistrare și toate atributele (ID_CLIENT și NUME) conțin valori atomice.
- În relația PARTICIPARE-SEDINTA există o cheie primară compusă formată din ID_ANGAJAT și ID_SEDINTA. Această cheie primară este un identificator unic și este valoare indivizibilă, deci relația PARTICIPARE_SEDINTA se află în FN1
- Pentru a exemplifica, vom lua entitățile ANGAJAT și TASK și presupunem că un angajat primește mai multe task-uri, deci mai multe ore de muncă.

ANGAJAT (#id_angajat)	Ore de muncă (durata unui task)
1	2h ,3.5h,6h
2	8.5h,4h
3	5h,5h,2.5h

Relația de mai sus nu se află în FN1 deoarece atributului DURATA a unui task nu îi corespunde o valoare indivizibila => Aplicam FN1

ANGAJAT (#id_angajat)	Ore de munca (durata unui task)
1	2h
1	3.5h
1	6h
2	8.5h
2	4h
3	5h
3	5h
3	2.5h

Asa arata tabelul dupa normalizare. Fiecărui atribut care compune relația îi corespunde o valoare indivizibilă.

★ FN2

FN2 impune condiții suplimentare față de FN1 pentru a evita dependențele funcționale parțiale sau tranzitive și pentru a asigura o structură mai robustă și mai eficientă pentru bazele de date.

Principalele cerințe ale FN2 sunt următoarele:

- Tabela trebuie să fie în Forma Normală 1 (FN1).
- Orice coloană care nu face parte din cheia primară trebuie să depindă de întreaga cheie primară și nu doar de o parte a acesteia.

În esență, FN2 are ca scop eliminarea redundanțelor și dependențelor funcționale care ar putea duce la anomalii în structura bazei de date. Prin îndeplinirea cerințelor FN2, se poate obține o bază de date mai eficientă, ușor de întreținut și fără pierdere de informații.

Pentru exemplificare, mă voi folosi de relația PARTICIPARE_SEDINTA.

#ID_ANGAJAT	NUME	#ID_SEDINTA	CONFIRMARE
1	Popescu	34	Confirmat
1	Manolache	56	Neconfirmat
1	Ispas	11	Neconfirmat
2	Preda	23	Confirmat
2	Mitroiu	14	Neconfirmat
3	Papuc	7	Confirmat

Un angajat poate participa la mai multe ședințe, iar o ședință poate avea mai mulți participanți.

După cum se observă, relația este în FN1 pentru că există identificator unic pentru toate intrările din tabel.
Pentru a face parte din FN2, atributele NUME și CONFIRMARE trebuie să depindă de întreaga cheie primară compusă: id_angajat# și id_sedinta#, dar se poate observa că acestea nu depind direct de toată cheia primară. Acest fapt se explică prin existența dependenței directe dintre #id_angajat și #id_sedinta. De aceea relația nu se află în FN2. De aici rezultă clasificarea următoarelor dependențe:

1. #ID_ANGAJAT —> Nume
2. #ID_ANGAJAT, #ID_SEDINTA —> Confirmare

Prin transformarea relației în FN2 vor rezulta 2 tabele:

#ID_ANGAJAT	NUME
1	Popescu
1	Manolache
1	Ispas
2	Preda
2	Mitroiu
3	Papuc

#ID_ANGAJAT	#ID_SEDINTA	CONFIRMARE
1	34	Confirmat
1	56	Neconfirmat
1	11	Neconfirmat
2	23	Confirmat
2	14	Neconfirmat
3	7	Confirmat

★ FN3

O relație se află în Forma Normală a Treia (FN3) atunci când îndeplinește următoarele cerințe:

- Tabela trebuie să fie în Forma Normală a Doua (FN2).
- Orice coloană care nu face parte din cheia primară trebuie să depindă de cheia primară în mod direct, și nu indirect prin intermediul altor coloane.

Cu alte cuvinte, FN3 elimină dependențele funcționale tranzitive, astfel încât nicio coloană care nu face parte din cheia primară să nu depindă de altă coloană care nu face parte din cheia primară.

Vom lua ca exemplu relația Birou care, fara FN3, ar arăta așa.

#ID_BIROU	CAPACITATE	ADRESA
1	3	123 Main Street
2	4	456 Elm Street
3	7	789 Oak Street
4	5	321 Pine Street
5	4	654 Maple Street

Se poate observa ca atributul ADRESA depinde de cheia primara ID_LOCATIE, iar din această cauză nu este in FN3.

Pentru a aduce la FN3 vom separa atributul referitor la locatie intr-un nou tabel LOCATIE. În locul atributului ADRESA vom pune cheia primara din noul tabel.

#ID_BIROU	CAPACITATE	ID_LOCATIE
1	3	1
2	4	4
3	7	7
4	5	3
5	4	6

ID_LOCATIE	ADRESA
1	123 Main Street
4	456 Elm Street
7	789 Oak Street
3	321 Pine Street
6	654 Maple Street

Mentionez faptul ca ID_LOCATIE este cheie străină în tabelul BIROU, astfel s-a realizat normalizarea la FN3.

10. Crearea unei secvente

```
CREATE SEQUENCE SEQ_PERIOADA
    INCREMENT BY 1
    START WITH 1
    MAXVALUE 9999
    NOCYCLE;
```

11.Creare si inserare

Tabelele au fost create (in partea dreapta se poate observa lista cu toate tabelele)

CLIENT:

```
CREATE TABLE CLIENT(
    ID_CLIENT NUMBER(4) PRIMARY KEY,
    NUME VARCHAR2(20) NOT NULL
);
```

	ID_CLIENT	NUME
1	987	John
2	234	Alice
3	512	Michael
4	789	Emma
5	345	William
6	111	Sophia
7	222	Oliver
8	333	Ava
9	444	Liam
10	555	Isabella

FIRMA:

```
CREATE TABLE FIRMA(
    CUI_FIRMA NUMBER(10)  PRIMARY KEY,
    NUME_FIRMA VARCHAR2(20) NOT NULL,
    COD_CAEN NUMBER(10) NOT NULL
);
```

	CUI_FIRMA	NUME_FIRMA	COD_CAEN
1	123456789	ABC Company	6201
2	987654321	XYZ Corporation	6202
3	456789123	DEF Ltd.	6203
4	654321987	PQR Solutions	6204
5	789123456	MNO Technologies	6205

CONTRACT:

```
CREATE TABLE CONTRACT(
        ID_CLIENT NUMBER(4),
        CUI_FIRMA NUMBER(10),
    PRIMARY KEY (ID_CLIENT, CUI_FIRMA),
    FOREIGN KEY (ID_CLIENT) REFERENCES CLIENT(ID_CLIENT),
    FOREIGN KEY (CUI_FIRMA) REFERENCES FIRMA(CUI_FIRMA)
);
```

13 rows

	ID_CLIENT	CUI_FIRMA
1	111	123456789
2	222	987654321
3	234	987654321
4	333	456789123
5	345	654321987
6	345	789123456
7	444	654321987
8	512	123456789
9	512	456789123
10	555	789123456
11	789	654321987
12	789	789123456
13	987	123456789

PERIOADA:

```
CREATE TABLE PERIOADA(
    ID_PERIOADA NUMBER(4),
    DATA_START DATE DEFAULT SYSDATE,
    DATA_FINISH DATE DEFAULT SYSDATE,
PRIMARY KEY (ID_PERIOADA)
);
```

	ID_PERIOADA	DATA_START	DATA_FINISH
1	1	2023-01-01	2023-01-31
2	2	2023-02-01	2023-02-28
3	3	2023-03-01	2023-03-31
4	4	2023-04-01	2023-04-30
5	5	2023-05-01	2023-05-31

LOCATIE:

```
CREATE TABLE LOCATIE(
    ID_LOCATIE NUMBER(4),
    ADRESA VARCHAR2(20),
    PRIMARY KEY (ID_LOCATIE)
);
```

5 rows

	ID_LOCATIE	ADRESA
1	1	123 Main Street
2	2	456 Elm Street
3	3	789 Oak Street
4	4	321 Pine Street
5	5	654 Maple Street

BIROU:

```
CREATE TABLE BIROU(
    ID_BIROU NUMBER(4),
    ID_LOCATIE NUMBER(4),
    CAPACITATE NUMBER(1),
    PRIMARY KEY (ID_BIROU),
    FOREIGN KEY (ID_LOCATIE) REFERENCES LOCATIE(ID_LOCATIE)
)
```

5 rows Tx: Manual

	ID_BIROU	ID_LOCATIE	CAPACITATE
1	1	5	5
2	2	4	4
3	3	3	7
4	4	2	5
5	5	1	4

ANGAJAT:

```
CREATE TABLE ANGAJAT(
     ID_ANGAJAT NUMBER(4),
     NUME VARCHAR2(10),
     PRENUME VARCHAR2(10),
     SALARIU NUMBER(6,2),
     DATA_NASTERII DATE,
      ID_BIROU NUMBER(4),
      PRIMARY KEY (ID_ANGAJAT),
      FOREIGN KEY (ID_BIROU) REFERENCES BIROU(ID_BIROU)
```

20 rows

	ID_ANGAJAT	NUME	PRENUME	SALARIU	DATA_NASTERII	ID_BIROU
1	1	Smith	Ken	5000.00	1990-05-15	4
2	2	Johnson	Emily	4500.00	1988-09-22	2
3	3	Brown	David	5200.00	1991-11-07	3
4	4	Wilson	Jessica	4800.00	1989-03-12	4
5	5	Miller	Andrew	5100.00	1992-07-29	2
6	6	Davis	Olivia	4900.00	1987-12-05	3
7	7	Taylor	William	5300.00	1993-10-18	1
8	8	Anderson	Sophia	4600.00	1991-02-03	2
9	9	Clark	Ethan	4800.00	1989-06-11	1
10	10	Walker	Ava	5100.00	1992-04-26	1
11	11	Smith	John	4800.00	1991-03-12	4
12	12	Johnson	Emma	5100.00	1988-07-29	5
13	13	Brown	Daniel	5200.00	1993-11-07	3
14	14	Wilson	Sophia	4800.00	1990-03-12	5
15	15	Miller	Matthew	5300.00	1987-07-29	5
16	16	Davis	Olivia	4900.00	1992-12-05	5
17	17	Taylor	Benjamin	5100.00	1991-10-18	3
18	18	Anderson	Isabella	4600.00	1989-02-03	1
19	19	Clark	Ethan	4800.00	1994-06-11	2
20	20	Walker	Mia	5100.00	1993-04-26	3

MANAGER:

```
CREATE TABLE MANAGER(
     ID_MANAGER NUMBER(4),
      NUME VARCHAR2(20),
     PRENUME VARCHAR2(20),
      DATA_NASTERII DATE DEFAULT SYSDATE,
      ANI_EXPERIENTA NUMBER(4),
      PRIMARY KEY (ID_MANAGER)
);
```

	ID_MANAGER	NUME	PRENUME	DATA_NASTERII	ANI_EXPERIENTA
1	1	Popescu	Ion	1985-01-01	6
2	2	Ionescu	Maria	1990-07-15	4
3	3	Constantin	Andrei	1988-03-10	8
4	4	Georgescu	Elena	1993-09-05	3
5	5	Popa	Mihai	1982-11-20	9

PROIECT:

```
CREATE TABLE PROIECT(
    ID_PROIECT NUMBER(4) PRIMARY KEY,
    NUME_PROIECT VARCHAR2(15),
    ID_MANAGER NUMBER(4),
    ID_PERIOADA NUMBER(4),
    CUI_FIRMA NUMBER(10),
    FOREIGN KEY (ID_MANAGER) REFERENCES MANAGER(ID_MANAGER),
    FOREIGN KEY (ID_PERIOADA) REFERENCES PERIOADA(ID_PERIOADA),
    FOREIGN KEY (CUI_FIRMA) REFERENCES FIRMA(CUI_FIRMA)
);
```

5 rows · Tx: Manual · DDL · CSV

	ID_PROIECT	NUME_PROIECT	ID_MANAGER	ID_PERIOADA	CUI_FIRMA
1	1	Iscoada	1	1	123456789
2	2	Stefanini Dev	2	2	987654321
3	3	Vinarte	3	3	456789123
4	4	CCS	4	4	654321987
5	5	Fashion House	5	5	789123456

TASK:

```
CREATE TABLE TASK(
    ID_TASK NUMBER(4),
    DEADLINE DATE,
    DIFICULTATE VARCHAR2(5),
    DURATA_ORE NUMBER(4,1),
    ID_ANGAJAT NUMBER(4),
    ID_PROIECT NUMBER(4),
    PRIMARY KEY (ID_TASK),
    FOREIGN KEY (ID_ANGAJAT) REFERENCES ANGAJAT(ID_ANGAJAT),
    FOREIGN KEY (ID_PROIECT) REFERENCES PROIECT(ID_PROIECT)
);
```

30 rows | Tx: Manual | DDL | CSV

	ID_TASK	DEADLINE	DIFICULTATE	DURATA_ORE	ID_ANGAJAT	ID_PROIECT
6	6	2023-05-27	GREU	7.8	13	3
7	7	2023-05-28	USOR	4.2	17	3
8	8	2023-05-29	MEDIU	5.9	11	2
9	9	2023-05-30	GREU	9.5	4	2
10	10	2023-05-31	USOR	2.8	9	4
11	11	2023-06-01	MEDIU	6.1	20	3
12	12	2023-06-02	GREU	8.4	12	5
13	13	2023-06-03	USOR	3.2	14	5
14	14	2023-06-04	MEDIU	5.7	15	5
15	15	2023-06-05	GREU	7.2	16	5
16	16	2023-06-06	USOR	2.9	15	5
17	17	2023-06-07	MEDIU	6.5	12	5
18	18	2023-06-08	GREU	8.8	14	5
19	19	2023-06-09	USOR	4.1	19	1
20	20	2023-06-10	MEDIU	5.6	16	5
21	21	2023-06-11	GREU	7.9	18	4
22	22	2023-06-12	USOR	3.8	9	4
23	23	2023-06-13	MEDIU	6.7	14	5
24	24	2023-06-14	GREU	9.2	5	1
25	25	2023-06-15	USOR	2.5	20	3
26	26	2023-06-16	MEDIU	5.3	2	1
27	27	2023-06-17	GREU	7.6	19	1
28	28	2023-06-18	USOR	3.7	8.0	1
29	29	2023-06-19	MEDIU	6.4	15	5
30	30	2023-06-20	GREU	8.7	19	1

SEDINTA:

```
CREATE TABLE SEDINTA(
    ID_SEDINTA NUMBER(4),
    ORA_START VARCHAR2(8),
    ORA_FINISH VARCHAR2(8),
    ID_MANAGER NUMBER(4),
    DATA DATE DEFAULT SYSDATE,
    PRIMARY KEY(ID_SEDINTA),
    FOREIGN KEY (ID_MANAGER) REFERENCES MANAGER(ID_MANAGER)
);
```

5 rows | Tx: Manual | DDL

	ID_SEDINTA	ORA_START	ORA_FINISH	ID_MANAGER	DATA
1	1	09:00	10:30	2	2023-05-22
2	2	14:00	15:30	3	2023-05-23
3	3	11:30	12:30	1	2023-05-24
4	4	16:00	17:30	5	2023-05-25
5	5	09:30	11:00	4	2023-05-26

PARTICIPARE_SEDINTA

```
CREATE TABLE PARTICIPARE_SEDINTA(
    ID_SEDINTA NUMBER(4),
    ID_ANGAJAT NUMBER(4),
    CONFIRMARE VARCHAR(20),
    FOREIGN KEY (ID_SEDINTA) REFERENCES SEDINTA(ID_SEDINTA),
    FOREIGN KEY(ID_ANGAJAT) REFERENCES ANGAJAT(ID_ANGAJAT)
)
```

18 rows | Tx: Manual

	ID_SEDINTA	ID_ANGAJAT	CONFIRMARE
1	3	2	CONFIRMAT
2	3	5	NECONFIRMAT
3	3	8	CONFIRMAT
4	3	19	CONFIRMAT
5	1	1	CONFIRMAT
6	1	4	NECONFIRMAT
7	1	11	CONFIRMAT
8	2	3	CONFIRMAT
9	2	6	NECONFIRMAT
10	2	13	CONFIRMAT
11	2	17	CONFIRMAT
12	2	20	NECONFIRMAT
13	5	7	CONFIRMAT
14	5	9	CONFIRMAT
15	4	12	NECONFIRMAT
16	4	14	CONFIRMAT
17	4	15	NECONFIRMAT
18	4	16	NECONFIRMAT

12. 5 CERERI SQL

1.Găsirea tuturor angajaților care lucrează la un proiect al unei anumite firme.

```sql
SELECT DISTINCT a.ID_ANGAJAT, a.NUME, a.PRENUME
FROM ANGAJAT a
    --SUBCERERI SINCRONIZATE IN CARE SUNT 3 TABELE
JOIN TASK t ON a.ID_ANGAJAT = t.ID_ANGAJAT
JOIN PROIECT p ON t.ID_PROIECT = p.ID_PROIECT
JOIN FIRMA f ON p.CUI_FIRMA = f.CUI_FIRMA
WHERE f.NUME_FIRMA = 'ABC Company';
```

2.Calculeaza orele de munca ale fiecarui angajat intr-o anumita perioada si spune daca rezonabil sau nu

```sql
--Grupări de date cu subcereri nesincronizate în care intervin cel puțin 3 tabele, funcții grup, filtrare la nivel de grupuri
--Utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE
WITH angajati_task AS (
SELECT a.ID_ANGAJAT, a.NUME, a.PRENUME, SUM(t.DURATA_ORE) AS DURATA_TOTALA
FROM ANGAJAT a
JOIN TASK t ON a.ID_ANGAJAT = t.ID_ANGAJAT
WHERE t.DEADLINE BETWEEN TO_DATE('01-01-2023', 'DD-MM-YYYY') AND TO_DATE('31-12-2023', 'DD-MM-YYYY')
GROUP BY a.ID_ANGAJAT, a.NUME, a.PRENUME
)
SELECT a.ID_ANGAJAT,
UPPER(a.NUME) AS UPPER_NUME,
INITCAP(a.PRENUME) AS INITCAP_PRENUME,
TO_CHAR(SYSDATE, 'DD-MM-YYYY') AS CURRENT_DATE,
TRUNC(t.DEADLINE) AS TRUNC_DEADLINE,
CASE WHEN a.DURATA_TOTALA > 10 THEN 'Exces de timp' ELSE 'Timp rezonabil' END AS DURATA_STATUS
FROM (
SELECT at.ID_ANGAJAT, at.NUME, at.PRENUME, at.DURATA_TOTALA,
ROW_NUMBER() OVER (PARTITION BY at.ID_ANGAJAT ORDER BY at.ID_ANGAJAT) AS rn
FROM angajati_task at
) a
JOIN TASK t ON a.ID_ANGAJAT = t.ID_ANGAJAT
WHERE a.rn = 1;
```

3.Verifica daca salariul a fost specificat la fiecare angajat

```sql
SELECT b.ID_BIROU, a.NUME, a.PRENUME,
       NVL(a.SALARIU, 0) AS SALARIU,
       DECODE(a.SALARIU, NULL, 'Nespecificat', 'Specificat') AS SALARIU_STATUS
FROM BIROU b
JOIN ANGAJAT a ON b.ID_BIROU = a.ID_BIROU
WHERE a.SALARIU = (
    SELECT MAX(SALARIU)
    FROM ANGAJAT
    WHERE ID_BIROU = b.ID_BIROU
);
```

4.Afiseaza numele si prenumele managerilor care au organizat cel putin o sedinta in care au participat mai mult de 3 angajati.

```sql
--Subcereri nesincronizate în clauza FROM
SELECT m.NUME, m.PRENUME
FROM (
    SELECT s.ID_MANAGER
    FROM SEDINTA s
    JOIN PARTICIPARE_SEDINTA ps ON s.ID_SEDINTA = ps.ID_SEDINTA
    GROUP BY s.ID_MANAGER, s.ID_SEDINTA
    HAVING COUNT(ps.ID_ANGAJAT) > 3
) subquery
JOIN MANAGER m ON subquery.ID_MANAGER = m.ID_MANAGER;
```

5.Să se afișeze numele și prenumele angajaților care lucrează în birourile cu capacitatea maximă și adresa biroului respectiv:

```sql
--Utilizarea a cel putin 1 bloc de cerere (clauza WITH):
WITH max_capacity AS (
    SELECT MAX(CAPACITATE) AS max_cap
    FROM BIROU
), angajati_birou AS (
    SELECT A.NUME, A.PRENUME, L.ADRESA, B.CAPACITATE
    FROM ANGAJAT A
    JOIN BIROU B ON A.ID_BIROU = B.ID_BIROU
    JOIN LOCATIE L ON B.ID_LOCATIE = L.ID_LOCATIE
)
SELECT AB.NUME, AB.PRENUME, AB.ADRESA
FROM angajati_birou AB
WHERE AB.CAPACITATE = (SELECT max_cap FROM max_capacity);
```

13.Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.

```sql
--Actualizarea adresei unei locatii pe baza numelui firmei cu care are contract:
UPDATE CLIENT
SET NUME = 'Catena'
WHERE ID_CLIENT = (
    SELECT ID_CLIENT
    FROM CONTRACT
    WHERE CUI_FIRMA = '123456789'
);

--Sterge participarea angajatului cu ID-ul 3 la toate sedintele.
DELETE FROM PARTICIPARE_SEDINTA
WHERE ID_ANGAJAT = 3
AND ID_SEDINTA IN (SELECT ID_SEDINTA FROM SEDINTA WHERE DATA < '26-JUL-2023');


--Schimba perioada unui proiect care are mai mult de 10 taskuri
UPDATE PROIECT
SET ID_PERIOADA = 5
WHERE ID_PROIECT IN (
    SELECT ID_PROIECT
    FROM TASK
    GROUP BY ID_PROIECT
    HAVING COUNT(*) > 10
);
```

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

Vizualizarea "VizualizareAngajatBirou" este creată prin intermediul unei interogări SELECT care combină tabelele "ANGAJAT" și "BIROU" și utilizează funcția DECODE pentru a transforma valorile din coloana "CAPACITATE" într-un mesaj asociat.

```
CREATE VIEW VizualizareAngajatBirou AS
SELECT a.NUME, a.PRENUME, a.SALARIU, b.ID_BIROU,
DECODE(b.CAPACITATE,
4, 'Birou mic',
5, 'Birou mediu',
7, 'Birou mare',
'Necunoscut') AS MESAJ_CAPACITATE
FROM ANGAJAT a
JOIN BIROU b ON a.ID_BIROU = b.ID_BIROU;
```

--operatie PERMISA

```
UPDATE VizualizareAngajatBirou
```

```
SET SALARIU = SALARIU + 500;
```

Această operație este permisă deoarece coloana "SALARIU" face parte din vizualizarea "VizualizareAngajatBirou" și este derivată direct din tabela "ANGAJAT". Prin urmare, putem modifica valorile acestei coloane în conformitate cu regulile de actualizare ale tabelei "ANGAJAT".

```
--operatie NEPERMISA

UPDATE VizualizareAngajatBirou
SET MESAJ_CAPACITATE = 'Birou extra-mare'
WHERE MESAJ_CAPACITATE = 'Birou mare';
```

Această operație este nepermisă deoarece coloana "MESAJ_CAPACITATE" nu face parte din nicio tabelă fizică din baza de date. Este o coloană calculată în cadrul vizualizării, bazată pe valorile din coloana "CAPACITATE" a tabelului "BIROU" și utilizând funcția DECODE. Deoarece aceasta nu este o coloană direct asociată cu o tabelă fizică, nu putem modifica valorile acesteia prin intermediul instrucțiunii UPDATE.

15.O cerere ce utilizează operația outer-join pe minimum 4 tabele,o cererece utilizează operația division și o cerere care implementează analiza top-n.

Cererea care utilizează operația **OUTER JOIN** pe minimum 4 tabele:

```
SELECT *
FROM CLIENT
LEFT JOIN CONTRACT ON CLIENT.ID_CLIENT = CONTRACT.ID_CLIENT
LEFT JOIN FIRMA ON CONTRACT.CUI_FIRMA = FIRMA.CUI_FIRMA
LEFT JOIN PROIECT ON CONTRACT.CUI_FIRMA = PROIECT.CUI_FIRMA
LEFT JOIN TASK ON PROIECT.ID_PROIECT = TASK.ID_PROIECT;
```

Această cerere utilizează operația OUTER JOIN pentru a obține toate înregistrările din tabela CLIENT și a le asocia cu înregistrările corespunzătoare din tabelele CONTRACT, FIRMA, PROIECT și TASK. Rezultatul va conține toate înregistrările din tabela CLIENT și înregistrările asociate din celelalte tabele, sau valorile NULL în cazul în care nu există o asociere.

Cererea care utilizează operația **DIVISION**:

```
SELECT DISTINCT ANGAJAT.ID_ANGAJAT, ANGAJAT.NUME, ANGAJAT.PRENUME
FROM ANGAJAT
WHERE NOT EXISTS (
    SELECT *
    FROM PROIECT
    WHERE NOT EXISTS (
```

```
        SELECT *
        FROM TASK
        WHERE TASK.ID_PROIECT = PROIECT.ID_PROIECT
        AND TASK.ID_ANGAJAT = ANGAJAT.ID_ANGAJAT
    )
);
```

Această cerere utilizează operația DIVISION pentru a obține toți angajații care sunt asignați tuturor proiectelor. Rezultatul va conține înregistrările din tabela ANGAJAT care îndeplinesc condiția că nu există niciun proiect pentru care angajatul să nu fie asignat la toate taskurile.

Cererea care implementează analiza **TOP-N:**

```
SELECT *
FROM (
    SELECT ANGAJAT.ID_ANGAJAT, ANGAJAT.NUME, ANGAJAT.PRENUME,
TASK.DURATA_ORE,
        ROW_NUMBER() OVER (ORDER BY TASK.DURATA_ORE DESC) AS rn
    FROM ANGAJAT
    INNER JOIN TASK ON ANGAJAT.ID_ANGAJAT = TASK.ID_ANGAJAT
) sub
WHERE rn <= 5;
```

Această cerere utilizează funcția analitică ROW_NUMBER() pentru a atribui un număr de ordine fiecărei înregistrări, ordonând rezultatele după durata orei taskului în ordine descrescătoare. Apoi, este selectat doar primele 5 înregistrări cu numerele de ordine mai mici sau egale cu 5, ceea ce reprezintă analiza TOP-N a celor mai lungi taskuri atribuite angajaților.

17.B.Aplicarea denormalizării, justificând necesitatea acesteia.

Pentru a aplica denormalizarea pe relația "PROIECT_TASK" din baza de date, putem crea o tabelă suplimentară denumită "PROIECT_TASK_COMPLET" care să conțină informații despre proiecte și task-urile asociate acestora. Aceasta simplifică structura de date și optimizează interogările care implică relația dintre proiecte și task-uri.

Structura tabelului "PROIECT_TASK_COMPLET" poate fi definită în felul următor:

```
CREATE TABLE PROIECT_TASK_COMPLET (
    ID_PROIECT NUMBER(4),
    NUME_PROIECT VARCHAR2(15),
    ID_MANAGER NUMBER(4),
    ID_PERIOADA NUMBER(4),
    CUI_FIRMA NUMBER(10),
    ID_TASK NUMBER(4),
```

```
    DEADLINE DATE,
    DIFICULTATE VARCHAR2(5),
    DURATA_ORE NUMBER(4,1),
    PRIMARY KEY (ID_PROIECT, ID_TASK),
    FOREIGN KEY (ID_PROIECT) REFERENCES PROIECT(ID_PROIECT),
    FOREIGN KEY (ID_TASK) REFERENCES TASK(ID_TASK)
);
```

În această tabelă, am inclus coloanele relevante referitoare la task-uri, cum ar fi deadline-ul, dificultatea și durata în ore. Aceste informații sunt denormalizate și duplicate în fiecare înregistrare, eliminând necesitatea de a face JOIN-uri între tabele separate pentru a obține informațiile complete despre task-uri în contextul unui proiect.

Astfel, atunci când avem nevoie să obținem detalii despre proiecte și task-uri, putem accesa direct tabela "**PROIECT_TASK_COMPLET**", evitând operațiile de JOIN pe tabele separate, ceea ce poate duce la o performanță îmbunătățită a interogărilor.

Prin adăugarea indexurilor corespunzătoare pe aceste coloane, putem accelera operațiile de căutare și filtrare în tabela denormalizată.

Denormalizarea în acest caz implică costul de a actualiza redundanțele datelor atunci când se fac modificări în tabelele originale (de exemplu, adăugarea sau ștergerea unui task sau actualizarea informațiilor despre un task).