Proiect Baze de Date – Firmă de contabilitate

Proiect realizat de Ivan Luca Cristian, Grupa 141

2025-2026, Sem. II

Ivan Luca-Cristian
Grupa 141

Cuprins

1. Descrierea modelului real, a utilității acestuia și a regulilor de funcționare 4

2. Prezentarea constrângerilor impuse asupra modelului 5

A. Constrângeri de integritate structurală 5

B. Constrângeri asupra datelor 5

3. Descrierea entităților, incluzând precizarea cheii primare 6

4. Descrierea relațiilor, precizând cardinalitatea acestora 8

5. Descrierea atributelor 9

6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5 13

7. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6. 14

8. Enumerarea schemelor relaționale 15

9. Realizarea normalizării până la FN3 16

A. Forma normală 1 (FN1) 16

B. Forma normală 2 (FN2) 16

C. Forma normală 3 (FN3) 16

10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele 17

11. Crearea tabelelor în SQL și inserarea de date 17

12. Formulați în limbaj natural și implementați 5 cereri SQL complexe 38

1. Pentru fiecare angajat care a efectuat activități, să se calculeze numărul total de ore lucrate conform pontajelor înregistrate. Să se afișeze doar acei angajați al căror volum total de ore cumulate depășește media generală a orelor introduse pe o singură înregistrare de pontaj. (c) 38

2. Să se afișeze detalii despre pontajele efectuate de angajați pentru clienți. Dacă descrierea activității nu a fost completată, se va afișa textul '[Activitatea nu este specificată]'. Să se clasifice regimul de lucru pe baza numărului de ore pontate astfel: dacă s-au pontat 8 ore se va afișa 'Full-time', dacă s-au pontat 6 ore se va afișa ‘Part-time extins’, dacă s-au pontat 4 ore se va afișa 'Part-time', iar pentru 2 ore se va afișa ‘Micro-job’; pentru orice alt număr de ore se va afișa 'Proiect/Peste program'. Să se ordoneze rezultatele descrescător după data pontajului și crescător după ID-ul angajatului. (d) .. 39

4. Să se afișeze codul unic și denumirea fiecărei firme cliente, numărul de contracte încheiate și suma totală pe care firma a achitat-o până în prezent, utilizând un tabel virtual ce pre-calculează prin agregare sumele plătite pentru fiecare factură. (b) 41

5. Să se afișeze denumirea serviciilor care au înregistrate pontaje în care numărul de ore lucrate este mai mare decât media orelor pontate pentru acel tip de serviciu la nivelul întregii firme, dar doar dacă

tariful negociat în cadrul contractului pentru clientul și serviciul respectiv este mai mare decât tariful lunar standard al serviciului. (a) 42

13. Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri 43

1. Se va majora cu 500 RON salariul tuturor angajaților (fără manager) care lucrează în departamentul numit 'Contabilitate Financiară'. 43

14. Crearea unei vizualizări complexe 45

15. Formulați în limbaj natural și implementați în SQL 3 cereri care să conțină: outer join pe minimum 4 tabele, division și analiză top-n 46

1. Să se afișeze o listă cu toți clienții din baza de date: numele firmelor, numărul contractelor semnate, denumirea serviciilor incluse în acele contracte și descrierea pontajelor efectuate. Raportul include absolut toți clienții (chiar dacă nu au contracte), toate contractele (chiar dacă nu au servicii asociate încă) și toate serviciile din contracte (chiar dacă nu s-a lucrat/pontat nimic pentru ele în luna curentă). (outer join) 46

2. Să se afișeze ID-ul, numele și prenumele angajaților care sunt atât de activi încât au efectuat pontaje pentru absolut toți clienții înregistrați în baza de date a firmei. (division) 47

16. Prezentarea planului de execuție a unei cereri complexe, optimizare/comparare plan alternativ folosind hint-uri si obiecte specifice optimizării cererilor 49

17. Realizarea normalizării BCNF, FN4, FN5. Aplicarea denormalizării, justificând necesitatea acesteia. 51

a) BCNF 51

b) FN4 52

c) FN5 53

d) De ce e utila denormalizarea? 54

18. Exemplificarea isolation levels prin exemple de tranzacții care se execută în paralel în condiții de concurență, evidențiind efectele diferitelor niveluri de izolare asupra concurenței și integrității datelor 56

19. Justificarea necesității/utilității migrării la o bază de date de tip NoSql 58

Ivan Luca-Cristian
Grupa 141

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

O aplicație pentru gestionarea activității unei firme de servicii contabile oferă posibilitatea de a monitoriza portofoliul de clienți, resursele umane, fluxul de documente și obligațiile fiscale ale acestora. Fiecare CLIENT este înregistrat în sistem printr-un identificator numeric unic (ID_CLIENT) generat prin secvență, având stocate denumirea firmei, codul unic de înregistrare fiscală (CUI) – ambele protejate prin constrângeri de unicitate – și adresa completă a sediului social.

Firma își organizează activitatea internă în DEPARTAMENTE specializate (Contabilitate Financiară, Salarizare și Resurse Umane, Consultanță Financiară etc.), fiecare departament fiind condus de un singur manager ales din rândul angajaților. În cadrul acestor structuri activează ANGAJAȚI (experți contabili, auditori, economiști), pentru care sistemul stochează detalii de contact (telefon, email), data angajării și salariul (protejat printr-o constrângere de verificare CHECK pentru a nu scădea sub minimul legal stabilit). Relația ierarhică de subordonare este modelată nativ în cadrul aceleiași tabele prin legătura reflexivă cu identificatorul managerului.

Colaborarea oficială cu beneficiarii este reglementată prin CONTRACTE, fiecare având un număr unic de înregistrare, o dată fixă de semnare și o legătură directă cu clientul beneficiar. Serviciile oferite (de la contabilitate financiară standard până la suport SAF-T) sunt definite în tabela SERVICII, care reține denumirea și tariful lunar de listă. Flexibilitatea comercială este asigurată prin tabela asociativă DETALII_CONTRACTE, care permite asocierea mai multor servicii unui singur contract și stocarea unui tarif negociat specific, ce poate suprascrie prețul standard.

Activitatea operațională zilnică și trasabilitatea muncii sunt asigurate prin monitorizarea a două fluxuri majore. Pe de o parte, entitatea DOCUMENTE_CLIENTI înregistrează actele primite spre procesare (facturi, extrase bancare, registre, contracte de muncă), consemnând cu precizie clientul de proveniență și angajatul care a operat documentul. Pe de altă parte, timpul efectiv de lucru este controlat prin PONTAJE – o relație asociativă complexă (ternară) ce corelează angajatul, clientul și tipul de serviciu prestat la o anumită dată, măsurând numărul de ore consumate (sistemul validând prin reguli stricte ca o sesiune de muncă să fie cuprinsă între 0 și maximum 12 ore).

Pentru asigurarea conformității legale în raport cu autoritățile statului, aplicația gestionează depunerea de DECLARAȚII fiscale (D112, D300, D406 etc.), urmărind termenele de scadență, clienții vizați și angajații responsabili de întocmire. Fiecare declarație este strâns legată de o RECIPISĂ oficială descărcată de pe portalul ANAF, sistemul monitorizând numărul de înregistrare al acesteia și statusul validării (Validă sau Respinsă cu erori).

Din punct de vedere financiar, în baza contractelor încheiate, firma emite periodic FACTURI individualizate prin serie și număr unic. Trasabilitatea fluxului de numerar și stingerea creanțelor sunt gestionate prin tabela PLĂȚI, care memorează suma încasată, data și timestamp-ul exact al tranzacției, precum și modalitatea de plată utilizată (Ordin de plată, Card bancar, Cash), asigurând controlul riguros asupra veniturilor firmei și a soldurilor clienților debitori.

2. Prezentarea constrângerilor impuse asupra modelului

A. Constrângeri de integritate structurală

Fiecare angajat trebuie să fie repartizat în mod obligatoriu unui singur departament. Un departament poate funcționa fără angajați înregistrați momentan sau poate avea mai mulți angajați.

Fiecare client este identificat în mod unic prin CUI (Cod Unic de Înregistrare). Un client poate avea unul sau mai multe contracte de prestări servicii active.

Un contract nu poate fi salvat în sistem fără a avea asociat cel puțin un serviciu prin intermediul tabelului DETALII_CONTRACTE.

O declarație fiscală poate exista în sistem fără o recipisă (în stadiul de "lucru"), dar o recipisă trebuie să fie obligatoriu atașată unei singure declarații depuse.

Orice document (factură, extras) introdus în sistem trebuie să indice în mod obligatoriu angajatul care l-a operat și clientul căruia îi aparține.

O înregistrare în tabelul PONTAJE este validă doar dacă conține simultan referințe către un angajat, un client și un serviciu prestat.

B. Constrângeri asupra datelor

Pentru fiecare client și angajat, adresa de e-mail și numărul de telefon trebuie să fie cunoscute.

Numărul contractului trebuie să fie unic la nivelul bazei de date.

Tariful serviciilor din SERVICII și valoarea facturilor din FACTURI trebuie să fie strict mai mari decât zero.

Data plății dintr-o înregistrare de PLATI nu poate fi anterioară datei emiterii facturii corespondente.

Numărul de ore pontate într-o singură înregistrare nu poate depăși 24 de ore.

Valoarea totală a unui document client din DOCUMENTE_CLIENTI trebuie să fie completată obligatoriu pentru a permite calculul indicatorilor fiscali.

Denumirea departamentului trebuie să fie unică (exemplu: nu pot exista două departamente cu numele "Audit").

Un departament este condus de un singur manager, ales din rândul angajaților firmei. Un angajat nu poate fi managerul mai multor departamente simultan.

Ivan Luca-Cristian
Grupa 141

3. Descrierea entităților, incluzând precizarea cheii primare

DEPARTAMENTE

- Reprezintă unitățile organizatorice ale firmei de contabilitate (ex: Gestiune, Salarizare).
- Cheie primară: **id_departament**

ANGAJATI

- Include persoanele care prestează servicii și operează documente în sistem.
- Cheie primară: **id_angajat**

CLIENTI

- Persoanele juridice sau fizice care beneficiază de serviciile contabile.
- Cheie primară: **id_client**

CONTRACTE

- Documentele juridice care reglementează relația dintre firmă și clienți.
- Cheie primară: **id_contract**

SERVICII

- Nomenclatorul de activități pe care firma le poate presta (ex: Audit, Expertiză).
- Cheie primară: **id_serviciu**

DETALII_CONTRACTE

- Entitate asociativă care specifică ce servicii sunt incluse în fiecare contract.
- Cheie primară: **id_contract**, **id_serviciu** (cheie compusă).

PONTAJE

- Entitate de legătură (ternară) pentru înregistrarea timpului lucrat de un angajat pentru un client pe un anumit serviciu.
- Cheie primară: **id_pontaj**

DOCUMENTE_CLIENTI

- Actele justificative (facturi, extrase) aduse de clienți spre prelucrare.
- Cheie primară: **id_document**

Ivan Luca-Cristian
Grupa 141

DECLARATII

- Formularele fiscale obligatorii ce trebuie depuse periodic pentru fiecare client.
- Cheie primară: **id_declaratie**

RECIPISE

- Dovezile de depunere electronică eliberate de autorități pentru declarațiile fiscale.
- Cheie primară: **id_recipisa**

FACTURI

- Documentele de plată emise de firma de contabilitate către clienții săi.
- Cheie primară: **id_factura**

PLATI

- Înregistrările tranzacțiilor financiare prin care clienții sting facturile emise.
- Cheie primară: **id_plata**

4. Descrierea relațiilor, precizând cardinalitatea acestora

RELAŢIE	**CARDINALITATE**	**Observaţii**
apartenență_angajat	Departament - Angajat: one-to-many	Un angajat aparține de un singur departament, în timp ce un departament are mai mulți angajați.
detalieri_contract	Contract - Serviciu: many-to-many	Un contract include mai multe servicii, iar un serviciu poate apărea în mai multe contracte. Se rezolvă prin tabelul asociativ DETALII_CONTRACTE.
alocare_portofoliu	Client - Angajat: many-to-many	Un client este gestionat de o echipă de angajați, iar un angajat are în portofoliu mai mulți clienți.
pontaj_activitate	Angajat - Client - Serviciu: many-to-many-to-many	Relație de aritate 3. Înregistrează cine (angajat), pentru cine (client) și ce (serviciu) a lucrat.
emitere_factura	Contract - Factură: one-to-many	În baza unui contract se pot emite mai multe facturi de-a lungul timpului.
stingere_factura	Factură - Plăți: one-to-many	O factură poate fi achitată integral sau prin mai multe plăți parțiale (tranșe).
operare_document	Angajat - Documente_Clienti: one-to-many	Un angajat este responsabil de introducerea și verificarea mai multor documente în sistem.
proveniență_acte	Client - Documente_Clienti: one-to-many	Un client furnizează firmei de contabilitate numeroase documente justificative pentru procesare.
întocmire_declarație	Client - Declarații: one-to-many	Pentru fiecare client se întocmesc lunar sau trimestrial mai multe declarații fiscale.
validare_depunere	Declarație - Recipise: one-to-one	O declarație depusă cu succes primește o singură recipisă oficială de validare.

5. Descrierea atributelor

DEPARTAMENTE

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_departament	NUMBER(5,0)	PK	1,2,3 etc.		
nume_departament	VARCHAR2(30)	NOT NULL, UNIQUE	Gestiune, Audit etc.		
id_manager	NUMBER(4,0)	FK			

ANGAJATI

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_angajat	NUMBER(4,0)	PK	1,2,3 etc.		
nume	VARCHAR2(50)	NOT NULL	Ion, Popescu etc.		
prenume	VARCHAR2(50)	NOT NULL	Maria, Andrei etc.		
telefon	CHAR(10)	NOT NULL	07...		
email	VARCHAR2(100)	NOT NULL	x@mail.com		
salariu	NUMBER(7,0)	NOT NULL			
data_angajarii	DATE	NOT NULL			
id_manager	NUMBER(4,0)	FK			
id_departament	NUMBER(5,0)	FK			

CLIENTI

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_client	NUMBER(7,0)	PK	1,2,3 etc.		
nume_firma	VARCHAR2(50)	NOT NULL, UNIQUE			
cui	VARCHAR2(11)	NOT NULL, UNIQUE	RO1234567		
adresa	VARCHAR2(100)	NOT NULL			

Ivan Luca-Cristian
Grupa 141

CONTRACTE

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_contract	NUMBER(10,0)	PK	1,2,3 etc.		
nr_contract	VARCHAR2(20)	NOT NULL, UNIQUE	CTR-YYYY-x		
data_semnare	DATE	NOT NULL			
id_client	VARCHAR2(100)	FK, NOT NULL			

SERVICII

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_serviciu	NUMBER(3,0)	PK	1,2,3 etc.		
denumire	VARCHAR2(100)	NOT NULL, UNIQUE			
tarif_lunar	NUMBER(7,0)	NOT NULL			

DETALII_CONTRACTE

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_contract	NUMBER(10,0)	PK, FK	1,2,3 etc.		
id_serviciu	NUMBER(3,0)	PK, FK			
tarif_negociat	NUMBER(7,0)				Clienții pot negocia tariful pe anumite servicii.

PONTAJE

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
data_pontaj	DATE	PK			
id_angajat	NUMBER(4,0)	PK, FK			
id_client	NUMBER(10,0)	PK, FK			
id_serviciu	NUMBER(3,0)	PK, FK			
nr_ore	NUMBER(2,1)	NOT NULL	>0, <=12		
descriere	VARCHAR2(200)				descriere muncă

Ivan Luca-Cristian
Grupa 141

DOCUMENTE_CLIENTI

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_document	NUMBER(10,0)	PK	1,2,3 etc.		
denumire	NUMBER(3,0)	NOT NULL			
tip	NUMBER(7,0)	NOT NULL			
id_client	NUMBER(10,0)	FK			
id_angajat	NUMBER(4,0)	FK			

DECLARATII

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_declaratie	NUMBER(10,0)	PK	1,2,3 etc.		
tip_declaratie	VARCHAR2(10)	NOT NULL			
data_scadenta	DATE	NOT NULL			
id_client	NUMBER(10,0)	FK			
id_angajat	NUMBER(4,0)	FK			

RECIPISE

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_recipisa	NUMBER(10,0)	PK	1,2,3 etc.		
id_declaratie	NUMBER(10,0)	FK			
nr_inregistrare	VARCHAR2(50)	NOT NULL UNIQUE			
status	VARCHAR2(30)	NOT NULL			
data_descarcare	DATE	NOT NULL		SYSDATE	

Ivan Luca-Cristian
Grupa 141

FACTURI

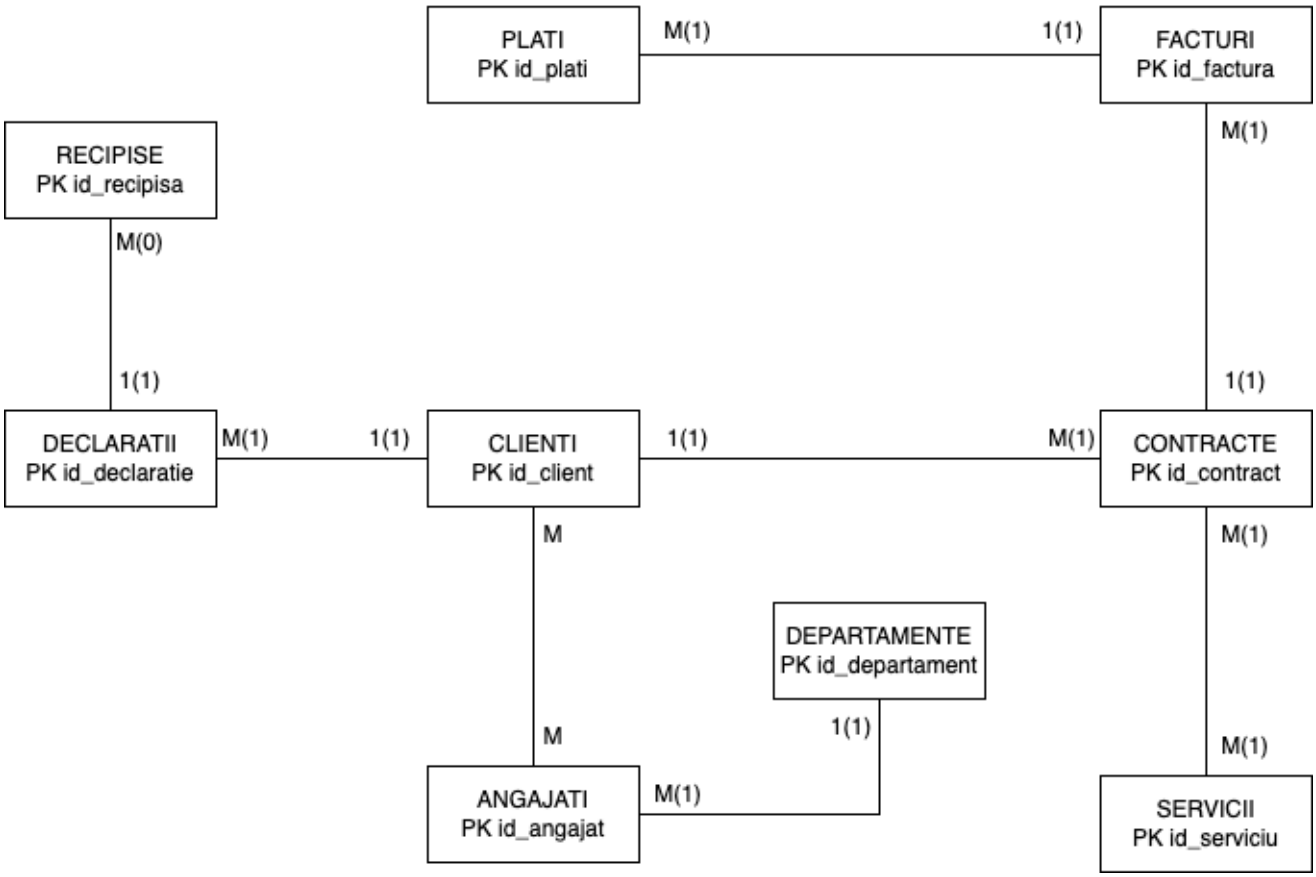
Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_factura	NUMBER(10,0)	PK	1,2,3 etc.		
serie_numar	VARCHAR2(50)	NOT NULL UNIQUE	INV-yyyy-x		
data_emitere	DATE	NOT NULL			
valoare	NUMBER(10,0)	NOT NULL			
id_contract	NUMBER(10,0)	NOT NULL FK			

PLATI

Atribut	Tip	Constrângeri	Valori posibile	Valoare implicită	Observații
id_plata	NUMBER(10,0)	PK	1,2,3 etc.		
suma	NUMBER(10,0)	NOT NULL			
data_plata	TIMESTAMP	NOT NULL			
mod_plata	VARCHAR2(30)	NOT NULL			
id_factura	VARCHAR2(100)	FK, NOT NULL			

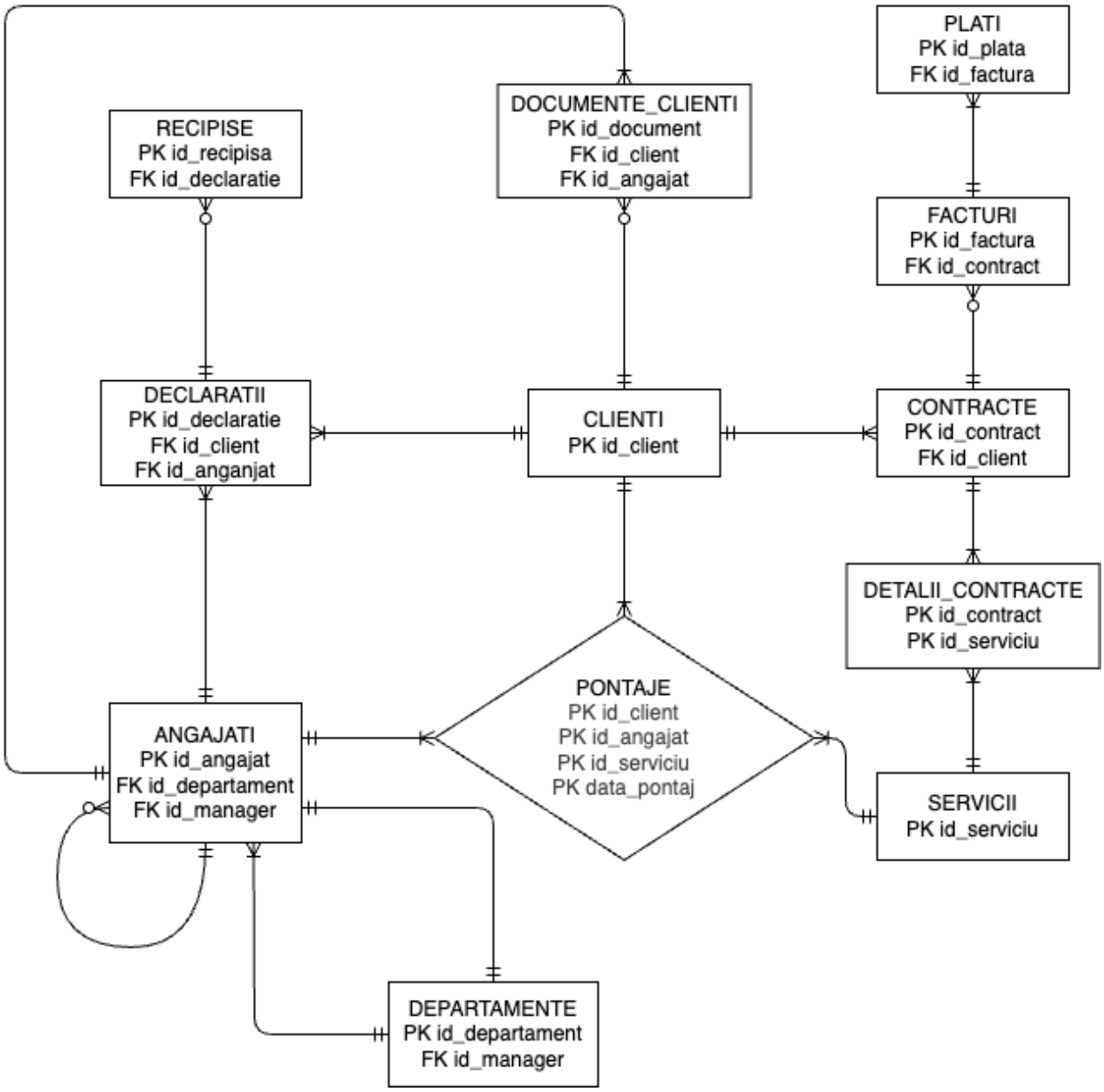
Ivan Luca-Cristian
Grupa 141

6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5



Ivan Luca-Cristian
Grupa 141

7. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6.



8. Enumerarea schemelor relaționale

DEPARTAMENTE (#id_departament, nume_departament, *id_manager*)
- *id_manager* face referință la **ANGAJATI** (id_angajat)

ANGAJATI (#id_angajat, nume, prenume, email, telefon, data_angajare, salariu, *id_manager id_departament*)
- *id_manager* face referință tot la **ANGAJATI** (id_angajat)
- *id_departament* face referință la **DEPARTAMENTE** (id_departament)

CLIENTI (#id_client, nume_firma, cui, adresa)

CONTRACTE (#id_contract, nr_contract, data_semnare, *id_client*)
- *id_client* face referință la **CLIENTI** (id_client)

SERVICII (#id_serviciu, denumire, tarif_lunar)

DETALII_CONTRACTE (#*id_contract*, #*id_serviciu*, tarif_negociat)
- *id_contract* face referință la **CONTRACTE** (id_contract)
- *id_serviciu* face referință la **SERVICII** (id_serviciu)

PONTAJE (#data_pontaj, #*id_angajat*, #*id_client*, #*id_serviciu*, nr_ore, descriere)
- *id_angajat* face referință la **ANGAJATI** (id_angajat)
- *id_client* face referință la **CLIENTI** (id_client)
- *id_serviciu* face referință la **SERVICII** (id_serviciu)

DOCUMENTE_CLIENTI (#id_document, denumire, tip, *id_client*, *id_angajat*)
- *id_client* face referință la **CLIENTI** (id_client)
- *id_angajat* face referință la **ANGAJATI** (id_angajat)

DECLARATII (#id_declaratie, tip_declaratie, data_scadenta, *id_client*, *id_angajat)*
- *id_client* face referință la **CLIENTI** (id_client)
- *id_angajat* face referință la **ANGAJATI** (id_angajat)

RECIPISE (#id_recipisa, *id_declaratie*, nr_inregistrare, status, data_descarcare)
- *id_declaratie* face referință la **DECLARATII** (id_declaratie)

FACTURI (#id_factura, serie_numar, data_emitere, valoare, *id_contract*)
- *id_contract* face referință la **CONTRACTE** (id_contract)

PLATI (#id_plata, suma, data_plata, mod_plata, *id_factura*)
- *id_factura* face referință la **FACTURI** (id_factura)

9. Realizarea normalizării până la FN3

A. Forma normală 1 (FN1)

O relație se află în FN1 dacă fiecărui atribut care o compune îi corespunde o valoare indivizabilă. Forma normală 1 este și cea care impune și faptul că fiecare înregistrare să fie definită astfel încât să fie identificată unic prin intermediul unei chei primare.

Exemplu non-FN1: CONTRACTE (id_contract, lista_servicii_incluse)

Problemă: Un contract poate include mai multe servicii (ex: "Audit, Salarizare, Consultanță"), deci atributul *lista_servicii_incluse* nu este atomic.

Descompunere: CONTRACTE (id_contract, data_semnare, id_client) - FN1
DETALII_CONTRACTE (id_contract, id_serviciu) - FN1

B. Forma normală 2 (FN2)

O relație se află în a doua formă normală dacă și numai dacă această relație este deja în FN1 și fiecare atribut care nu este cheie primară este dependent de întreaga cheie primară.

FN2 interzice existența dependențelor funcționale parțiale în cadrul relației.

Dacă unul sau mai multe elemente sunt dependente funcțional numai de o parte a cheii primare, atunci ele trebuie sa fie separate în tabele diferite. Dacă tabela are o cheie primară formată din numai un atribut, atunci ea este automat în FN2.

Exemplu non-FN2: DETALII_CONTRACTE (id_contract, id_serviciu, data_alocare, tarif_ora_serviciu)

Dependență: *tarif_ora_serviciu* depinde doar de *id_serviciu*, nu și de *id_contract*.

Descompunere: SERVICII (id_serviciu, tarif_ora_serviciu) - FN2
DETALII_CONTRACTE (id_contract, id_serviciu, data_alocare) - FN2

C. Forma normală 3 (FN3)

O relație este în a treia formă normală dacă și numai dacă este in FN2 și fiecare atribut care nu este cheie depinde direct de cheia primară.

Exemplu non-FN3: ANGAJATI (id_angajat, nume, id_departament, nume_departament)

Dependență: *nume_departament* depinde de *id_departament*, care la rândul lui depinde de *id_angajat*.

Descompunere: DEPARTAMENTE (id_departament, nume_departament) - FN3
ANGAJATI (id_angajat, nume, id_departament) - FN3

Ivan Luca-Cristian
Grupa 141

10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele

+

11. Crearea tabelelor în SQL și inserarea de date

```
CREATE TABLE CLIENTI (
ID_CLIENT NUMBER(7,0) PRIMARY KEY,
NUME_FIRMA VARCHAR2(50) NOT NULL UNIQUE,
CUI NUMBER(10,0) NOT NULL UNIQUE,
ADRESA VARCHAR2(100) NOT NULL);

CREATE SEQUENCE SEQ_CLIENTI START WITH 1 INCREMENT BY 1;

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'NOVA PROMPT SRL', '15366582','Aleea Mizil 60, 032345 Bucuresti');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'ARGENTINE BAR ' || CHR(38) || ' RESTAURANT SRL', '12409034',
'Strada Traian 129, 024041 Bucuresti');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'KRONOS CONSULTING SRL', '31544892', 'Bulevardul Eroilor 45,
500007 Brasov');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'NEXUS LOGISTIC TRANSPORTER SRL', '18965412', 'Soseaua
Borsului 12A, 410605 Oradea');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'DELTA AGRO TRADING SRL', '42155763', 'Calea Moldova 180, 710012
Botosani');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'VIP MEDICA PREST SRL', '35698411', 'Bulevardul Tomis 234, 900501
Constanta');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'BRAVE IT SOFTWARE SRL', '28944125', 'Strada Zorilor 10, 400600
Cluj-Napoca');
```

```
INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'ZETA CONSTRUCT ENGINEERING SRL', '14755896', 'Calea Mosilor 21, 020851 Bucuresti');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'PRIME DISTRIBUTION NET SRL', '39855412', 'Strada Cutitelor 4, 200450 Craiova');

INSERT INTO CLIENTI (ID_CLIENT, NUME_FIRMA, CUI, ADRESA)
VALUES (SEQ_CLIENTI.nextval, 'SMART RETAIL CONCEPT SRL', '22415896', 'Bulevardul Revolutiei 85, 310130 Arad');
```

	ID_CLIENT	NUME_FIRMA	CUI	ADRESA
1	1	NOVA PROMPT SRL	15366582	Aleea Mizil 60, 032345 Bucuresti
2	2	ARGENTINE BAR & RESTAURANT SRL	12409034	Strada Traian 129, 024041 Bucuresti
3	3	KRONOS CONSULTING SRL	31544892	Bulevardul Eroilor 45, 500007 Brasov
4	4	NEXUS LOGISTIC TRANSPORTER SRL	18965412	Soseaua Borsului 12A, 410605 Oradea
5	6	DELTA AGRO TRADING SRL	42155763	Calea Moldova 180, 710012 Botosani
6	7	VIP MEDICA PREST SRL	35698411	Bulevardul Tomis 234, 900501 Constanta
7	8	BRAVE IT SOFTWARE SRL	28944125	Strada Zorilor 10, 400600 Cluj-Napoca
8	9	ZETA CONSTRUCT ENGINEERING SRL	14755896	Calea Mosilor 21, 020851 Bucuresti
9	10	PRIME DISTRIBUTION NET SRL	39855412	Strada Cutitelor 4, 200450 Craiova
10	11	SMART RETAIL CONCEPT SRL	22415896	Bulevardul Revolutiei 85, 310130 Arad

Se creează tabelele **DEPARTAMENTE** și **ANGAJATI**. Din cauza dependenței circulare, trebuie să nu punem *id_manager* din **DEPARTAMENTE** încă FK, dar după ce este creată și tabela **ANGAJATI**, modificăm tabelul **DEPARTAMENTE**, adăugând o constrângere astfel:

```
CREATE TABLE DEPARTAMENTE(
ID_DEPARTAMENT NUMBER(5,0) PRIMARY KEY,
NUME_DEPARTAMENT VARCHAR2(30) NOT NULL UNIQUE,
ID_MANAGER NUMBER(4,0) UNIQUE);

CREATE TABLE ANGAJATI(
ID_ANGAJAT NUMBER(4,0) PRIMARY KEY,
NUME VARCHAR2(50) NOT NULL,
PRENUME VARCHAR2(50) NOT NULL,
TELEFON CHAR(10) NOT NULL,
```

```
EMAIL VARCHAR2(100) NOT NULL,
SALARIU NUMBER(7,0) NOT NULL,
DATA_ANGAJARII DATE NOT NULL,
ID_MANAGER NUMBER(4,0) REFERENCES ANGAJATI(ID_ANGAJAT),
ID_DEPARTAMENT NUMBER(5,0) REFERENCES DEPARTAMENTE(ID_DEPARTAMENT));

ALTER TABLE DEPARTAMENTE
ADD CONSTRAINT ID_MANAGER FOREIGN KEY (ID_MANAGER) REFERENCES
ANGAJATI(ID_ANGAJAT);

--CONDITIE CA SALARIUL SA FIE MAI MARE DECAT MINIMUL PE ECONOMIE
ALTER TABLE ANGAJATI ADD CONSTRAINT SALARIU CHECK (SALARIU > 2574);

CREATE SEQUENCE SEQ_DEPARTMENT START WITH 10 INCREMENT BY 10;

INSERT INTO DEPARTAMENTE (ID_DEPARTAMENT, NUME_DEPARTAMENT, ID_MANAGER)
VALUES (SEQ_DEPARTMENT.nextval, 'Contabilitate Financiara', NULL);

INSERT INTO DEPARTAMENTE (ID_DEPARTAMENT, NUME_DEPARTAMENT, ID_MANAGER)
VALUES (SEQ_DEPARTMENT.nextval, 'Salarizare si Resurse Umane', NULL);

INSERT INTO DEPARTAMENTE (ID_DEPARTAMENT, NUME_DEPARTAMENT, ID_MANAGER)
VALUES (SEQ_DEPARTMENT.nextval, 'Consultanta Financiara', NULL);

INSERT INTO DEPARTAMENTE (ID_DEPARTAMENT, NUME_DEPARTAMENT, ID_MANAGER)
VALUES (SEQ_DEPARTMENT.nextval, 'Audit si Certificare', NULL);

INSERT INTO DEPARTAMENTE (ID_DEPARTAMENT, NUME_DEPARTAMENT, ID_MANAGER)
VALUES (SEQ_DEPARTMENT.nextval, 'Contabilitate de Gestiune', NULL);

CREATE SEQUENCE SEQ_ANGAJATI START WITH 101 INCREMENT BY 1;

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU,
DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Grant', 'Che', '0730745708', 'ajfromthelane@gmail.com',
30000, TO_DATE('01-03-2020', 'DD-MM-YYYY'), NULL, NULL);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU,
DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Anghel', 'Bogdan', '0745748364', 'ian@gmail.com', 15000,
TO_DATE('06-05-2021', 'DD-MM-YYYY'), 100, NULL);
```

```
INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Nedelcu', 'Edward', '0727845282', 'azteca@gmail.com', 16160, TO_DATE('12-03-2022', 'DD-MM-YYYY'), 100, NULL);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Iliescu', 'Renato', '0769246677', 'renvto@gmail.com', 13500, TO_DATE('03-09-2025', 'DD-MM-YYYY'), 100, NULL);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Capraru', 'Stefan', '0792764272', 'oscar@gmail.com', 12000 , TO_DATE('01-08-2023', 'DD-MM-YYYY'), 100, NULL);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Matasaru', 'Leonard', '0727948165', 'mgl@gmail.com', 11000, TO_DATE('23-07-2022', 'DD-MM-YYYY'), 100, NULL);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Avram', 'Stefan', '0772462781', 'nane@gmail.com', 6000, TO_DATE('09-09-2024', 'DD-MM-YYYY'), 101, 10);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Hasnas', 'Calin', '0772583422', 'calinacho@gmail.com', 4500, TO_DATE('12-10-2025', 'DD-MM-YYYY'), 101, 10);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Pantic', 'Alberto', '0720138485', 'abili@gmail.com', 7800, TO_DATE('02-12-2023', 'DD-MM-YYYY'), 101, 10);


INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Mihalache', 'Andrei', '0713910317', 'andreimgk@gmail.com', 5000, TO_DATE('02-03-2024', 'DD-MM-YYYY'), 102, 20);
```

```
INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Musat', 'Daniel', '0728762712', 'nelithegod@gmail.com', 9000, TO_DATE('23-06-2021', 'DD-MM-YYYY'), 102, 20);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Miscoci', 'Luca', '0755222046', 'miscellaneous@gmail.com', 5600, TO_DATE('14-12-2022', 'DD-MM-YYYY'), 103, 30);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Mosu', 'David', '0730745708', 'luk@gmail.com', 6767, TO_DATE('01-02-2025', 'DD-MM-YYYY'), 103, 30);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Berechet', 'Matei', '0797164732', 'berechet@gmail.com', 3800, TO_DATE('29-11-2025', 'DD-MM-YYYY'), 104, 40);

INSERT INTO ANGAJATI (ID_ANGAJAT, NUME, PRENUME, TELEFON, EMAIL, SALARIU, DATA_ANGAJARII, ID_MANAGER, ID_DEPARTAMENT)
VALUES (SEQ_ANGAJATI.nextval, 'Thies', 'Elias', '0723826842', 'eliasflex@gmail.com', 5900, TO_DATE('02-02-2021', 'DD-MM-YYYY'), 105, 50);

UPDATE DEPARTAMENTE
SET ID_MANAGER = 101
WHERE ID_DEPARTAMENT = 10;

UPDATE DEPARTAMENTE
SET ID_MANAGER = 102
WHERE ID_DEPARTAMENT = 20;

UPDATE DEPARTAMENTE
SET ID_MANAGER = 103
WHERE ID_DEPARTAMENT = 30;

UPDATE DEPARTAMENTE
SET ID_MANAGER = 104
WHERE ID_DEPARTAMENT = 40;
```

Ivan Luca-Cristian
Grupa 141

```
UPDATE DEPARTAMENTE
SET ID_MANAGER = 105
WHERE ID_DEPARTAMENT = 50;

COMMIT;
```

	ID_DEPARTAMENT	NUME_DEPARTAMENT	ID_MANAGER
1	10	Contabilitate Financiara	101
2	20	Salarizare si Resurse Umane	102
3	30	Consultanta Financiara	103
4	40	Audit si Certificare	104
5	50	Contabilitate de Gestiune	105

	ID_ANGAJAT	NUME	PRENUME	TELEFON	EMAIL	SALARIU	DATA_ANGAJARII	ID_MANAGER	ID_DEPARTAMENT
1	100	Grant	Che	0730745708	ajfromthelane@gmail.com	30000	01-MAR-20	(null)	(null)
2	101	Anghel	Bogdan	0745748364	ian@gmail.com	15000	06-MAY-21	100	10
3	102	Nedelcu	Edward	0727845282	azteca@gmail.com	16160	12-MAR-22	100	20
4	103	Iliescu	Renato	0769246677	renvto@gmail.com	13500	03-SEP-25	100	30
5	104	Capraru	Stefan	0792764272	oscar@gmail.com	12000	01-AUG-23	100	40
6	105	Matasaru	Leonard	0727948165	mgl@gmail.com	11000	23-JUL-22	100	50
7	106	Avram	Stefan	0772462781	nane@gmail.com	6000	09-SEP-24	101	10
8	107	Hasnas	Calin	0772583422	calinacho@gmail.com	4500	12-OCT-25	101	10
9	108	Pantic	Alberto	0720138485	abili@gmail.com	7800	02-DEC-23	101	10
10	109	Mihalache	Andrei	0713910317	andreimgk@gmail.com	5000	02-MAR-24	102	20
11	110	Musat	Daniel	0728762712	nelithegod@gmail.com	9000	23-JUN-21	102	20
12	111	Miscoci	Luca	0755222046	miscellaneous@gmail.com	5600	14-DEC-22	103	30
13	112	Mosu	David	0730745708	luk@gmail.com	6767	01-FEB-25	103	30
14	113	Berechet	Matei	0797164732	berechet@gmail.com	3800	29-NOV-25	104	40
15	114	Thies	Elias	0723826842	eliasflex@gmail.com	5900	02-FEB-21	105	50

Ivan Luca-Cristian
Grupa 141

```
CREATE TABLE CONTRACTE(
ID_CONTRACT NUMBER(10,0) PRIMARY KEY,
NR_CONTRACT VARCHAR2(20) UNIQUE NOT NULL,
DATA_SEMNARE DATE NOT NULL,
ID_CLIENT NUMBER(7,0) NOT NULL REFERENCES CLIENTI);

CREATE SEQUENCE SEQ_CONTRACTE START WITH 1 INCREMENT BY 1;

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-1', TO_DATE('01-05-2026', 'DD-MM-YYYY'), 1);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-2', TO_DATE('03-05-2026', 'DD-MM-YYYY'), 2);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-3', TO_DATE('12-05-2026', 'DD-MM-YYYY'), 2);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-4', TO_DATE('13-05-2026', 'DD-MM-YYYY'), 3);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-5', TO_DATE('21-05-2026', 'DD-MM-YYYY'), 5);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-6', TO_DATE('21-05-2026', 'DD-MM-YYYY'), 6);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-7', TO_DATE('22-05-2026', 'DD-MM-YYYY'), 6);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-8', TO_DATE('22-05-2026', 'DD-MM-YYYY'), 7);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-9', TO_DATE('08-05-2026', 'DD-MM-YYYY'), 8);

INSERT INTO CONTRACTE (ID_CONTRACT, NR_CONTRACT, DATA_SEMNARE, ID_CLIENT)
VALUES (SEQ_CONTRACTE.nextval, 'CTR-2026-10', TO_DATE('16-05-2026', 'DD-MM-YYYY'),
9);
```

Ivan Luca-Cristian
Grupa 141

	ID_CONTRACT	NR_CONTRACT	DATA_SEMNARE	ID_CLIENT
1	1	CTR-2026-1	01-MAY-26	1
2	2	CTR-2026-2	03-MAY-26	2
3	3	CTR-2026-3	12-MAY-26	2
4	4	CTR-2026-4	13-MAY-26	3
5	5	CTR-2026-5	21-MAY-26	5
6	6	CTR-2026-6	21-MAY-26	6
7	7	CTR-2026-7	22-MAY-26	6
8	8	CTR-2026-8	22-MAY-26	7
9	9	CTR-2026-9	08-MAY-26	8
10	10	CTR-2026-10	16-MAY-26	9

```
CREATE TABLE SERVICII(
ID_SERVICIU NUMBER(3,0) PRIMARY KEY,
DENUMIRE VARCHAR2(100) NOT NULL UNIQUE,
TARIF_LUNAR NUMBER(7,0) NOT NULL);

CREATE SEQUENCE SEQ_SERVICII START WITH 1 INCREMENT BY 1;

INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Contabilitate Financiara', 700);

INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Salarizare si Gestiune HR', 600);


INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Consultanta Financiara Permanenta', 950);

INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Contabilitate de Gestiune si Management', 1200);

INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Suport si Monitorizare SAF-T', 450);

INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Pachet Start-Up (contabilitate completa)', 350);
```

Ivan Luca-Cristian
Grupa 141

```
INSERT INTO SERVICII (ID_SERVICIU, DENUMIRE, TARIF_LUNAR)
VALUES (SEQ_SERVICII.nextval, 'Pachet Corporate (Full-Service)', 2000);
```

	ID_SERVICIU	DENUMIRE	TARIF_LUNAR
1	1	Contabilitate Financiara	700
2	2	Salarizare si Gestiune HR	600
3	3	Consultanta Financiara Permanenta	950
4	4	Contabilitate de Gestiune si Management	1200
5	5	Suport si Monitorizare SAF-T	450
6	6	Pachet Start-Up (contabilitate completa)	350
7	7	Pachet Corporate (Full-Service)	2000

```
CREATE TABLE DETALII_CONTRACTE(
ID_CONTRACT NUMBER(10,0) REFERENCES CONTRACTE(ID_CONTRACT),
ID_SERVICIU NUMBER(3,0) REFERENCES SERVICII(ID_SERVICIU),
TARIF_NEGOCIAT NUMBER(7,0),
CONSTRAINT DETALII_CONTRACTE PRIMARY KEY (ID_CONTRACT, ID_SERVICIU));

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (1, 1, NULL);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (1, 2, 550);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (2, 3, NULL);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (2, 4, 1100);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (2, 5, 450);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (3, 4, 1125);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (3, 5, NULL);
```

Ivan Luca-Cristian
Grupa 141

```
INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (4, 7, 1900);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (5, 5, NULL);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (5, 2, 575);

INSERT INTO DETALII_CONTRACTE (ID_CONTRACT, ID_SERVICIU, TARIF_NEGOCIAT)
VALUES (5, 4, NULL);
```

	ID_CONTRACT	ID_SERVICIU	TARIF_NEGOCIAT
1	1	1	(null)
2	1	2	550
3	2	3	(null)
4	3	4	1125
5	4	7	1900
6	5	5	(null)
7	5	2	575
8	2	4	1100
9	2	5	450
10	3	5	(null)

```
CREATE TABLE PONTAJE(
DATA_PONTAJ DATE NOT NULL,
ID_ANGAJAT NUMBER(4,0) REFERENCES ANGAJATI(ID_ANGAJAT),
ID_CLIENT NUMBER(10,0) REFERENCES CLIENTI(ID_CLIENT),
ID_SERVICIU NUMBER(3,0) REFERENCES SERVICII(ID_SERVICIU),
NR_ORE NUMBER (2,1) NOT NULL,
DESCRIERE VARCHAR2(200),

CONSTRAINT PONTAJE PRIMARY KEY (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT,
ID_SERVICIU),
```

```
-- NU ARE SENS CA CINEVA SA MUNCEASCA 0 ORE, IAR UN ANGAJAT NU POATE LUCRA DINTR-ODATA MAI MULT DE 12 ORE
CONSTRAINT ORE_MUNCA CHECK (NR_ORE > 0 AND NR_ORE <=12));

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('20-05-2026', 'DD-MM-YYYY'), 108, 2, 3, 7, NULL);

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('21-05-2026', 'DD-MM-YYYY'), 108, 2, 3, 6, NULL);

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('21-05-2026', 'DD-MM-YYYY'), 114, 3, 4, 8, 'Gestionarea resurselor financiare');

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('22-05-2026', 'DD-MM-YYYY'), 107, 4, 7, 8, 'Contabilitate financiara');

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('22-05-2026', 'DD-MM-YYYY'), 111, 4, 7, 8, 'Consultanta');

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('22-05-2026', 'DD-MM-YYYY'), 109, 5, 2, 6, 'Suport salarizare');

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('23-05-2026', 'DD-MM-YYYY'), 109, 5, 2, 4, NULL);

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('23-05-2026', 'DD-MM-YYYY'), 107, 1, 1, 8, 'Contabilitate financiara');

INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE, DESCRIERE)
VALUES (TO_DATE('23-05-2026', 'DD-MM-YYYY'), 112, 2, 5, 8, 'Monitorizare');
```

Ivan Luca-Cristian
Grupa 141

```
INSERT INTO PONTAJE (DATA_PONTAJ, ID_ANGAJAT, ID_CLIENT, ID_SERVICIU, NR_ORE,
DESCRIERE)
VALUES (TO_DATE('23-05-2026', 'DD-MM-YYYY'), 108, 2, 3, 8, NULL);
```

	DATA_PONTAJ	ID_ANGAJAT	ID_CLIENT	ID_SERVICIU	NR_ORE	DESCRIERE
1	20-MAY-26	108	2	3	7	(null)
2	21-MAY-26	108	2	3	6	(null)
3	21-MAY-26	114	3	4	8	Gestionarea resurselor financiare
4	22-MAY-26	107	4	7	8	Contabilitate financiara
5	22-MAY-26	111	4	7	8	Consultanta
6	22-MAY-26	109	5	2	6	Suport salarizare
7	23-MAY-26	109	5	2	4	(null)
8	23-MAY-26	107	1	1	8	Contabilitate financiara
9	23-MAY-26	112	2	5	8	Monitorizare
10	23-MAY-26	108	2	3	8	(null)

```
CREATE TABLE DOCUMENTE_CLIENTI(
ID_DOCUMENT NUMBER(10,0) PRIMARY KEY,
DENUMIRE VARCHAR2(100) NOT NULL,
TIP VARCHAR2(30) NOT NULL,
ID_CLIENT NUMBER(10,0) NOT NULL REFERENCES CLIENTI(ID_CLIENT),
ID_ANGAJAT NUMBER(4,0) NOT NULL REFERENCES ANGAJATI(ID_ANGAJAT));

CREATE SEQUENCE SEQ_DOCUMENTE_CLIENTI START WITH 1 INCREMENT BY 1;

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Facturi Furnizori Aprilie 2026', 'Factura', 1, 107);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Extras de Cont BCR Mai 2026', 'Extras Bancar',
1, 107);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Registru de Casa Zilele 1-15', 'Registru', 2, 108);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Stat de Plata Personal Aprilie', 'Salarizare', 3,
108);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Facturi Clienti Emise Semestrul 1', 'Factura', 4,
109);
```

Ivan Luca-Cristian
Grupa 141

```
INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Deconturi Cheltuieli Transport', 'Decont', 5, 110);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Bonuri Fiscale Combustibil', 'Bon Fiscal', 2, 111);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Contract de Munca', 'Contract', 3, 112);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Avize de Insotire a Marfii Marfuri', 'Aviz', 1, 113);

INSERT INTO DOCUMENTE_CLIENTI (id_document, denumire, tip, id_client, id_angajat)
VALUES (SEQ_DOCUMENTE_CLIENTI.nextval, 'Chitante Utilitati Sediu Central', 'Chitanta', 4,
114);
```

	ID_DOCUMENT	DENUMIRE	TIP	ID_CLIENT	ID_ANGAJAT
1	1	Facturi Furnizori Aprilie	Factura	1	107
2	2	Extras de Cont BCR Aprilie	Extras Bancar	1	107
3	3	Registru de Casa	Registru	2	108
4	4	Stat de Plata Personal Aprilie	Salarizare	3	108
5	5	Facturi Clienti Emise Semestrul 1	Factura	4	109
6	6	Deconturi Cheltuieli Transport	Decont	5	110
7	7	Bonuri Fiscale Combustibil	Bon Fiscal	2	111
8	8	Contract de Munca	Contract	3	112
9	9	Avize de Insotire a Marfii	Aviz	1	113
10	10	Chitante Utilitati Sediu Central	Chitanta	4	114

```
CREATE TABLE DECLARATII(
ID_DECLARATIE NUMBER(10,0) PRIMARY KEY,
TIP_DECLARATIE VARCHAR2(10) NOT NULL,
DATA_SCADENTA DATE NOT NULL,
ID_CLIENT NUMBER(10,0) NOT NULL REFERENCES CLIENTI(ID_CLIENT),
ID_ANGAJAT NUMBER(4,0) NOT NULL REFERENCES ANGAJATI(ID_ANGAJAT));

CREATE SEQUENCE SEQ_DECLARATII START WITH 1 INCREMENT BY 1;

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA,
ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D112', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 1, 107);
```

```
INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D300', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 1, 107);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D112', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 2, 108);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D406', TO_DATE('31-05-2026', 'DD-MM-YYYY'), 2, 108);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D112', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 3, 112);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D100', TO_DATE('25-07-2026', 'DD-MM-YYYY'), 3, 109);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D112', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 4, 114);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D300', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 4, 111);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D112', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 5, 110);

INSERT INTO DECLARATII (ID_DECLARATIE, TIP_DECLARATIE, DATA_SCADENTA, ID_CLIENT, ID_ANGAJAT)
VALUES (SEQ_DECLARATII.nextval, 'D100', TO_DATE('25-05-2026', 'DD-MM-YYYY'), 5, 113);
```

Ivan Luca-Cristian
Grupa 141

	ID_DECLARATIE	TIP_DECLARATIE	DATA_SCADENTA	ID_CLIENT	ID_ANGAJAT
1	1	D112	25-MAY-26	1	107
2	2	D300	25-MAY-26	1	107
3	3	D112	25-MAY-26	2	108
4	4	D406	31-MAY-26	2	108
5	5	D112	25-MAY-26	3	112
6	6	D100	25-JUL-26	3	109
7	7	D112	25-MAY-26	4	114
8	8	D300	25-MAY-26	4	111
9	9	D112	25-MAY-26	5	110
10	10	D100	25-MAY-26	5	113

```
CREATE TABLE RECIPISE(
ID_RECIPISA NUMBER(10,0) PRIMARY KEY,
ID_DECLARATIE NUMBER(10,0) NOT NULL REFERENCES DECLARATII(ID_DECLARATIE),
NR_INREGISTRARE VARCHAR2(50) NOT NULL,
STATUS VARCHAR2(30) NOT NULL,
DATA_DESCARCARE DATE DEFAULT SYSDATE);

CREATE SEQUENCE SEQ_RECIPISE START WITH 1 INCREMENT BY 1;

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS,
DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 1, 'ANAF-ERR-49102', 'Respinsa cu erori', TO_DATE('20-05-
2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS,
DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 1, 'ANAF-OK-49155', 'Valida', TO_DATE('21-05-2026', 'DD-
MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS,
DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 2, 'ANAF-OK-49201', 'Valida', TO_DATE('22-05-2026', 'DD-
MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS,
DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 3, 'ANAF-OK-49312', 'Valida', TO_DATE('24-05-2026', 'DD-
MM-YYYY'));
```

```
INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 4, 'ANAF-ERR-49400', 'Respinsa cu erori', TO_DATE('24-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 5, 'ANAF-OK-49588', 'Valida', TO_DATE('18-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 6, 'ANAF-ERR-49601', 'Respinsa cu erori', TO_DATE('19-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 6, 'ANAF-ERR-49602', 'Respinsa cu erori', TO_DATE('20-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 6, 'ANAF-OK-49655', 'Valida', TO_DATE('21-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 7, 'ANAF-OK-49700', 'Valida', TO_DATE('22-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 8, 'ANAF-OK-49811', 'Valida', TO_DATE('23-05-2026', 'DD-MM-YYYY'));

INSERT INTO RECIPISE (ID_RECIPISA, ID_DECLARATIE, NR_INREGISTRARE, STATUS, DATA_DESCARCARE)
VALUES (SEQ_RECIPISE.nextval, 9, 'ANAF-OK-49944', 'Valida', TO_DATE('23-05-2026', 'DD-MM-YYYY'));
```

Ivan Luca-Cristian
Grupa 141

	ID_RECIPISA	ID_DECLARATIE	NR_INREGISTRARE	STATUS	DATA_DESCARCARE
1	1	1	ANAF-ERR-49102	Respinsa cu erori	20-MAY-26
2	2	1	ANAF-OK-49155	Valida	21-MAY-26
3	3	2	ANAF-OK-49201	Valida	22-MAY-26
4	4	3	ANAF-OK-49312	Valida	24-MAY-26
5	5	4	ANAF-ERR-49400	Respinsa cu erori	24-MAY-26
6	6	5	ANAF-OK-49588	Valida	18-MAY-26
7	7	6	ANAF-ERR-49601	Respinsa cu erori	19-MAY-26
8	8	6	ANAF-ERR-49602	Respinsa cu erori	20-MAY-26
9	9	6	ANAF-OK-49655	Valida	21-MAY-26
10	10	7	ANAF-OK-49700	Valida	22-MAY-26
11	11	8	ANAF-OK-49811	Valida	23-MAY-26
12	12	9	ANAF-OK-49944	Valida	23-MAY-26

```
CREATE TABLE FACTURI (
ID_FACTURA   NUMBER(10,0) PRIMARY KEY,
SERIE_NUMAR  VARCHAR2(50) NOT NULL UNIQUE,
DATA_EMITERE DATE NOT NULL,
VALOARE      NUMBER(10,0) NOT NULL,
ID_CONTRACT  NUMBER(10,0) NOT NULL REFERENCES CONTRACTE(ID_CONTRACT));

CREATE SEQUENCE SEQ_FACTURI START WITH 1 INCREMENT BY 1;

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-001', TO_DATE('02-05-2026', 'DD-MM-YYYY'),
1300, 1);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-002', TO_DATE('04-05-2026', 'DD-MM-YYYY'), 950,
2);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-003', TO_DATE('13-05-2026', 'DD-MM-YYYY'),
1125, 3);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
```

```
VALUES (SEQ_FACTURI.nextval, 'INV-2026-004', TO_DATE('15-05-2026', 'DD-MM-YYYY'),
1900, 4);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-005', TO_DATE('22-05-2026', 'DD-MM-YYYY'),
1025, 5);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-006', TO_DATE('15-02-2026', 'DD-MM-YYYY'),
3500, 3);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-007', TO_DATE('15-03-2026', 'DD-MM-YYYY'),
3500, 3);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-008', TO_DATE('28-02-2026', 'DD-MM-YYYY'),
1200, 4);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-009', TO_DATE('31-03-2026', 'DD-MM-YYYY'),
1200, 4);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-010', TO_DATE('31-03-2026', 'DD-MM-YYYY'),
4000, 5);

INSERT INTO FACTURI (ID_FACTURA, SERIE_NUMAR, DATA_EMITERE, VALOARE,
ID_CONTRACT)
VALUES (SEQ_FACTURI.nextval, 'INV-2026-011', TO_DATE('30-04-2026', 'DD-MM-YYYY'),
4000, 5);
```

Ivan Luca-Cristian
Grupa 141

	ID_FACTURA	SERIE_NUMAR	DATA_EMITERE	VALOARE	ID_CONTRACT
1	1	INV-2026-001	02-MAY-26	1300	1
2	2	INV-2026-002	04-MAY-26	950	2
3	3	INV-2026-003	13-MAY-26	1125	3
4	4	INV-2026-004	15-MAY-26	1900	4
5	5	INV-2026-005	22-MAY-26	1025	5
6	6	INV-2026-006	15-FEB-26	3500	3
7	7	INV-2026-007	15-MAR-26	3500	3
8	8	INV-2026-008	28-FEB-26	1200	4
9	9	INV-2026-009	31-MAR-26	1200	4
10	10	INV-2026-010	31-MAR-26	4000	5
11	11	INV-2026-011	30-APR-26	4000	5

```
CREATE TABLE PLATI (
ID_PLATA NUMBER(10,0) PRIMARY KEY,
SUMA NUMBER(10,0) NOT NULL,
DATA_PLATA TIMESTAMP NOT NULL,
MOD_PLATA VARCHAR2(30) NOT NULL,
ID_FACTURA NUMBER(10,0) NOT NULL REFERENCES FACTURI(ID_FACTURA),

   CONSTRAINT SUMA_POZITIVA CHECK (SUMA > 0)
);

CREATE SEQUENCE SEQ_PLATI START WITH 1 INCREMENT BY 1;

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES  (SEQ_PLATI.nextval,  1300,  TO_TIMESTAMP('06-05-2026  10:30:00',  'DD-MM-YYYY
HH24:MI:SS'), 'Ordin de plata', 1);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES  (SEQ_PLATI.nextval,  1300,  TO_TIMESTAMP('06-05-2026  10:30:00',  'DD-MM-YYYY
HH24:MI:SS'), 'Ordin de plata', 1);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES  (SEQ_PLATI.nextval,  950,  TO_TIMESTAMP('10-05-2026  14:15:00',  'DD-MM-YYYY
HH24:MI:SS'), 'Card bancar', 2);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
```

```
VALUES (SEQ_PLATI.nextval, 1125, TO_TIMESTAMP('18-05-2026 09:00:00', 'DD-MM-YYYY HH24:MI:SS'), 'Ordin de plata', 3);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 1000, TO_TIMESTAMP('19-05-2026 11:45:22', 'DD-MM-YYYY HH24:MI:SS'), 'Ordin de plata', 4);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 900, TO_TIMESTAMP('22-05-2026 16:20:00', 'DD-MM-YYYY HH24:MI:SS'), 'Chitanta cash', 4);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 3500, TO_TIMESTAMP('20-02-2026 13:00:00', 'DD-MM-YYYY HH24:MI:SS'), 'Ordin de plata', 6);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 3500, TO_TIMESTAMP('25-03-2026 15:30:45', 'DD-MM-YYYY HH24:MI:SS'), 'Ordin de plata', 7);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 1200, TO_TIMESTAMP('05-03-2026 10:00:00', 'DD-MM-YYYY HH24:MI:SS'), 'Chitanta cash', 8);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 1200, TO_TIMESTAMP('07-04-2026 12:15:00', 'DD-MM-YYYY HH24:MI:SS'), 'Ordin de plata', 9);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 4000, TO_TIMESTAMP('10-04-2026 17:00:00', 'DD-MM-YYYY HH24:MI:SS'), 'Card bancar', 10);

INSERT INTO PLATI (ID_PLATA, SUMA, DATA_PLATA, MOD_PLATA, ID_FACTURA)
VALUES (SEQ_PLATI.nextval, 4000, TO_TIMESTAMP('05-05-2026 11:10:30', 'DD-MM-YYYY HH24:MI:SS'), 'Ordin de plata', 11);
```

Ivan Luca-Cristian
Grupa 141

	ID_PLATA	SUMA	DATA_PLATA	MOD_PLATA	ID_FACTURA
1	1	1300	06-MAY-26 10.30.00.000000000 AM	Ordin de plata	1
2	2	1300	06-MAY-26 10.30.00.000000000 AM	Ordin de plata	1
3	3	950	10-MAY-26 02.15.00.000000000 PM	Card bancar	2
4	4	1125	18-MAY-26 09.00.00.000000000 AM	Ordin de plata	3
5	5	1000	19-MAY-26 11.45.22.000000000 AM	Ordin de plata	4
6	6	900	22-MAY-26 04.20.00.000000000 PM	Chitanta cash	4
7	7	3500	20-FEB-26 01.00.00.000000000 PM	Ordin de plata	6
8	8	3500	25-MAR-26 03.30.45.000000000 PM	Ordin de plata	7
9	9	1200	05-MAR-26 10.00.00.000000000 AM	Chitanta cash	8
10	10	1200	07-APR-26 12.15.00.000000000 PM	Ordin de plata	9
11	11	4000	10-APR-26 05.00.00.000000000 PM	Card bancar	10
12	12	4000	05-MAY-26 11.10.30.000000000 AM	Ordin de plata	11

Ivan Luca-Cristian
Grupa 141

12. Formulați în limbaj natural și implementați 5 cereri SQL complexe

1. *Pentru fiecare angajat care a efectuat activități, să se calculeze numărul total de ore lucrate conform pontajelor înregistrate. Să se afișeze doar acei angajați al căror volum total de ore cumulate depășește media generală a orelor introduse pe o singură înregistrare de pontaj. (c)*

```
SELECT ID_ANGAJAT, SUM(NR_ORE) AS "Total ore lucrate" , COUNT(*) AS "Total acțiuni"
FROM PONTAJE
GROUP BY ID_ANGAJAT
HAVING SUM(NR_ORE) > (SELECT AVG(NR_ORE)
                      FROM pontaje);
```

```
HAVING SUM(NR_ORE) > (SELECT AVG(NR_ORE) as Medie -- filtrare la nivel de gru
                      FROM pontaje);
```

Query Result

SQL | All Rows Fetched: 1 in 0.007 seconds

	MEDIE
1	7.1

```
SELECT ID_ANGAJAT, SUM(NR_ORE) AS "Total ore lucrate" , COUNT(*) AS "Total actiuni"
FROM PONTAJE
GROUP BY ID_ANGAJAT  --functie grup
HAVING SUM(NR_ORE) > (SELECT AVG(NR_ORE) as Medie -- filtrare la nivel de grupuri
                      FROM pontaje);
```

Query Result

SQL | All Rows Fetched: 6 in 0.008 seconds

	ID_ANGAJAT	Total ore lucrate	Total acțiuni
1	108	21	3
2	114	8	1
3	107	16	2
4	111	8	1
5	109	10	2
6	112	8	1

2. *Să se afișeze detalii despre pontajele efectuate de angajați pentru clienți. Dacă descrierea activității nu a fost completată, se va afișa textul '[Activitatea nu este specificată]'. Să se clasifice regimul de lucru pe baza numărului de ore pontate astfel: dacă s-au pontat 8 ore se va afișa 'Full-time', dacă s-au pontat 6 ore se va afișa 'Part-time extins', dacă s-au pontat 4 ore se va afișa 'Part-time', iar pentru 2 ore se va afișa 'Micro-job'; pentru orice alt număr de ore se va afișa 'Proiect/Peste program'. Să se ordoneze rezultatele descrescător după data pontajului și crescător după ID-ul angajatului. (d)*

```
SELECT ID_ANGAJAT, ID_CLIENT, DATA_PONTAJ,
DECODE(NR_ORE, 8, 'Full-time',
               6, 'Part-time extins',
               4, 'Part-time',
               2, 'Micro-Job',
               'Proiect/Peste program') AS REGIM,
NVL(DESCRIERE, '[Activitatea nu este specificată]') AS DESCRIERE
FROM PONTAJE
ORDER BY DATA_PONTAJ DESC, ID_ANGAJAT ASC;
```

```
SELECT ID_ANGAJAT, ID_CLIENT, DATA_PONTAJ,
DECODE(NR_ORE, 8, 'Full-time',
               6, 'Part-time extins',
               4, 'Part-time',
               2, 'Micro-Job',
               'Proiect/Peste program') AS REGIM ,
       NVL(DESCRIERE, '[Activitatea nu este specificată]') AS DESCRIERE
FROM PONTAJE
ORDER BY DATA_PONTAJ DESC, ID_ANGAJAT ASC;
```

Query Result

SQL | All Rows Fetched: 10 in 0.004 seconds

	ID_ANGAJAT	ID_CLIENT	DATA_PONTAJ	REGIM	DESCRIERE
1	107	1	23-MAY-26	Full-time	Contabilitate financiara
2	108	2	23-MAY-26	Full-time	[Activitatea nu este specificată]
3	109	5	23-MAY-26	Part-time	[Activitatea nu este specificată]
4	112	2	23-MAY-26	Full-time	Monitorizare
5	107	4	22-MAY-26	Full-time	Contabilitate financiara
6	109	5	22-MAY-26	Part-time extins	Suport salarizare
7	111	4	22-MAY-26	Full-time	Consultanta
8	108	2	21-MAY-26	Part-time extins	[Activitatea nu este specificată]
9	114	3	21-MAY-26	Full-time	Gestionarea resurselor financiare
10	108	2	20-MAY-26	Proiect/Peste program	[Activitatea nu este specificată]

Ivan Luca-Cristian
Grupa 141

3. *Să se extragă date din tabela de angajați. Să se transforme numele de familie în majuscule, să se extragă primele 3 caractere din adresa de email, să se determine anul angajării, numărul de luni de la data angajării până în prezent și să se stabilească o clasificare dinamică a fidelității angajatului în funcție de numărul de zile de vechime. (e+f)*

```
WITH ANALIZA_ANGAJATI AS (SELECT ID_ANGAJAT, NUME,
                          SUBSTR(EMAIL, 1, 3) AS PREFIX_EMAIL,
                          EXTRACT(YEAR FROM DATA_ANGAJARII) AS AN_ANGAJARE,
                          MONTHS_BETWEEN(SYSDATE, DATA_ANGAJARII) AS LUNI_VECHIME,
                          TRUNC(SYSDATE) - TRUNC(DATA_ANGAJARII) AS ZILE_LUCRATE
                          FROM ANGAJATI)

SELECT ID_ANGAJAT, UPPER(NUME) AS NUME, PREFIX_EMAIL, AN_ANGAJARE,
ROUND(LUNI_VECHIME, 1) AS "LUNI IN COMPANIE",
        CASE
          WHEN ZILE_LUCRATE <= 731 THEN 'Junior'
          WHEN ZILE_LUCRATE BETWEEN 731 AND 1827 THEN 'Mid-Level'
          ELSE 'Senior'
        END AS VECHIME
FROM ANALIZA_ANGAJATI;
```

```
WITH ANALIZA_ANGAJATI AS (SELECT ID_ANGAJAT, NUME,
                          SUBSTR(EMAIL, 1, 3) AS PREFIX_EMAIL,
                          EXTRACT(YEAR FROM DATA_ANGAJARII) AS AN_ANGAJARE,
                          MONTHS_BETWEEN(SYSDATE, DATA_ANGAJARII) AS LUNI_VECHIME,
                          TRUNC(SYSDATE) - TRUNC(DATA_ANGAJARII) AS ZILE_LUCRATE
                          FROM ANGAJATI)

SELECT ID_ANGAJAT, UPPER(NUME) AS NUME, PREFIX_EMAIL, AN_ANGAJARE, ROUND(LUNI_VECHIME, 1) AS "LUNI IN COMPANIE",
        CASE
            WHEN ZILE_LUCRATE <= 731 THEN 'Junior'
            WHEN ZILE_LUCRATE BETWEEN 731 AND 1827 THEN 'Mid-Level'
            ELSE 'Senior'
        END AS VECHIME
FROM ANALIZA_ANGAJATI;
```

Query R...

SQL | All Rows Fetched: 15 in 0.007 seconds

	ID_ANGAJAT	NUME	PREFIX_EMAIL	AN_ANGAJARE	LUNI IN COMPANIE	VECHIME
1	100	GRANT	ajf	2020	74.8	Senior
2	101	ANGHEL	ian	2021	60.6	Senior
3	102	NEDELCU	azt	2022	50.4	Mid-Level
4	103	ILIESCU	ren	2025	8.7	Junior
5	104	CAPRARU	osc	2023	33.8	Mid-Level
6	105	MATASARU	mgl	2022	46.1	Mid-Level
7	106	AVRAM	nan	2024	20.5	Junior
8	107	HASNAS	cal	2025	7.4	Junior
9	108	PANTIC	abi	2023	29.7	Mid-Level
10	109	MIHALACHE	and	2024	26.7	Mid-Level
11	110	MUSAT	nel	2021	59.1	Mid-Level
12	111	MISCOCI	mis	2022	41.3	Mid-Level
13	112	MOSU	luk	2025	15.8	Junior
14	113	BERECHET	ber	2025	5.9	Junior
15	114	THIES	eli	2021	63.7	Senior

4. *Să se afișeze codul unic și denumirea fiecărei firme cliente, numărul de contracte încheiate și suma totală pe care firma a achitat-o până în prezent, utilizând un tabel virtual ce pre-calculează prin agregare sumele plătite pentru fiecare factură. (b)*

```
SELECT C.ID_CLIENT, C.NUME_FIRMA,
    COUNT(CON.ID_CONTRACT) AS NUMAR_CONTRACTE,
    SUM(NVL(CALC_PLATI.TOTAL_ACHITAT, 0)) AS TOTAL_PLATIT_CLIENT
FROM CLIENTI C
LEFT JOIN CONTRACTE CON ON C.ID_CLIENT = CON.ID_CLIENT
LEFT JOIN FACTURI F ON CON.ID_CONTRACT = F.ID_CONTRACT
LEFT JOIN (SELECT ID_FACTURA, SUM(SUMA) AS TOTAL_ACHITAT
        FROM PLATI
        GROUP BY ID_FACTURA) CALC_PLATI ON F.ID_FACTURA = CALC_PLATI.ID_FACTURA
GROUP BY C.ID_CLIENT, C.NUME_FIRMA;
```

```
SELECT C.ID_CLIENT, C.NUME_FIRMA,
       COUNT(CON.ID_CONTRACT) AS NUMAR_CONTRACTE,
       SUM(NVL(CALC_PLATI.TOTAL_ACHITAT, 0)) AS TOTAL_PLATIT_CLIENT
FROM CLIENTI C
LEFT JOIN CONTRACTE CON ON C.ID_CLIENT = CON.ID_CLIENT
LEFT JOIN FACTURI F ON CON.ID_CONTRACT = F.ID_CONTRACT
LEFT JOIN (SELECT ID_FACTURA, SUM(SUMA) AS TOTAL_ACHITAT
           FROM PLATI
           GROUP BY ID_FACTURA) CALC_PLATI ON F.ID_FACTURA = CALC_PLATI.ID_FACTURA
GROUP BY C.ID_CLIENT, C.NUME_FIRMA;
```

Query Result

SQL | All Rows Fetched: 10 in 0.045 seconds

	ID_CLIENT	NUME_FIRMA	NUMAR_CONTRACTE	TOTAL_PLATIT_CLIENT
1	1	NOVA PROMPT SRL	1	2600
2	2	ARGENTINE BAR & RESTAURANT SRL	4	9075
3	3	KRONOS CONSULTING SRL	3	4300
4	5	DELTA AGRO TRADING SRL	3	8000
5	6	VIP MEDICA PREST SRL	2	0
6	7	BRAVE IT SOFTWARE SRL	1	0
7	9	PRIME DISTRIBUTION NET SRL	1	0
8	4	NEXUS LOGISTIC TRANSPORTER SRL	0	0
9	10	SMART RETAIL CONCEPT SRL	0	0
10	8	ZETA CONSTRUCT ENGINEERING SRL	1	0

Ivan Luca-Cristian
Grupa 141

5. *Să se afişeze denumirea serviciilor care au înregistrate pontaje în care numărul de ore lucrate este mai mare decât media orelor pontate pentru acel tip de serviciu la nivelul întregii firme, dar doar dacă tariful negociat în cadrul contractului pentru clientul şi serviciul respectiv este mai mare decât tariful lunar standard al serviciului. (a)*

```
SELECT S.DENUMIRE
FROM SERVICII S
WHERE EXISTS (SELECT 1
          FROM PONTAJE P
          WHERE P.ID_SERVICIU = S.ID_SERVICIU
          AND P.NR_ORE > (SELECT AVG(P2.NR_ORE)
                    FROM PONTAJE P2
                    WHERE P2.ID_SERVICIU = S.ID_SERVICIU)
    AND P.ID_CLIENT IN (
      SELECT DC.ID_CONTRACT
      FROM DETALII_CONTRACTE DC
      JOIN CONTRACTE C ON DC.ID_CONTRACT = C.ID_CONTRACT
      WHERE DC.ID_SERVICIU = S.ID_SERVICIU
       AND DC.TARIF_NEGOCIAT > S.TARIF_LUNAR
       AND C.ID_CLIENT = P.ID_CLIENT));
```

```
SELECT S.DENUMIRE
FROM SERVICII S
WHERE EXISTS (SELECT 1
              FROM PONTAJE P
              WHERE P.ID_SERVICIU = S.ID_SERVICIU
              AND P.NR_ORE > (SELECT AVG(P2.NR_ORE)
                              FROM PONTAJE P2
                              WHERE P2.ID_SERVICIU = S.ID_SERVICIU)
                              AND P.ID_CLIENT IN
                              (SELECT DC.ID_CONTRACT
                               FROM DETALII_CONTRACTE DC
                               JOIN CONTRACTE C ON DC.ID_CONTRACT = C.ID_CONTRACT
                               WHERE DC.ID_SERVICIU = S.ID_SERVICIU
                               AND DC.TARIF_NEGOCIAT > S.TARIF_LUNAR
                               AND C.ID_CLIENT = P.ID_CLIENT));
```

Query Result

SQL | All Rows Fetched: 0 in 0.024 seconds

DENU...

Ivan Luca-Cristian
Grupa 141

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

1. *Se va majora cu 500 RON salariul tuturor angajaților (fără manager) care lucrează în departamentul numit 'Contabilitate Financiară'.*

```
UPDATE ANGAJATI
SET SALARIU = SALARIU + 500
WHERE ID_DEPARTAMENT = (SELECT ID_DEPARTAMENT
                        FROM DEPARTAMENTE
                        WHERE  UPPER(NUME_DEPARTAMENT)  =  'CONTABILITATE  FINANCIARA')
AND  ID_MANAGER != 100;
```

	ID_ANGAJAT	NUME	PRENUME	TELEFON	EMAIL	SALARIU	DATA_ANGAJARII	ID_MANAGER	ID_DEPARTAMENT
1	100	Grant	Che	0730745708	ajfromthelane@gmail.com	30000	01-MAR-20	(null)	(null)
2	101	Anghel	Bogdan	0745748364	ian@gmail.com	15000	06-MAY-21	100	10
3	102	Nedelcu	Edward	0727845282	azteca@gmail.com	16160	12-MAR-22	100	20
4	103	Iliescu	Renato	0769246677	renvto@gmail.com	13500	03-SEP-25	100	30
5	104	Capraru	Stefan	0792764272	oscar@gmail.com	12000	01-AUG-23	100	40
6	105	Matasaru	Leonard	0727948165	mgl@gmail.com	11000	23-JUL-22	100	50
7	106	Avram	Stefan	0772462781	nane@gmail.com	6500	09-SEP-24	101	10
8	107	Hasnas	Calin	0772583422	calinacho@gmail.com	5000	12-OCT-25	101	10
9	108	Pantic	Alberto	0720138485	abili@gmail.com	8300	02-DEC-23	101	10
10	109	Mihalache	Andrei	0713910317	andreimgk@gmail.com	5000	02-MAR-24	102	20
11	110	Musat	Daniel	0728762712	nelithegod@gmail.com	9000	23-JUN-21	102	20
12	111	Miscoci	Luca	0755222046	miscellaneous@gmail.com	5600	14-DEC-22	103	30
13	112	Mosu	David	0730745708	luk@gmail.com	6767	01-FEB-25	103	30
14	113	Berechet	Matei	0797164732	berechet@gmail.com	3800	29-NOV-25	104	40
15	114	Thies	Elias	0723826842	eliasflex@gmail.com	5900	02-FEB-21	105	50

2. *Se vor șterge toate declarațiile care nu au recipisă de la ANAF.*

```
DELETE FROM DECLARATII
WHERE ID_DECLARATIE NOT IN (SELECT ID_DECLARATIE
                            FROM RECIPISE
                            WHERE ID_DECLARATIE IS NOT NULL);
```

	ID_DECLARATIE	TIP_DECLARATIE	DATA_SCADENTA	ID_CLIENT	ID_ANGAJAT
1	1	D112	25-MAY-26	1	107
2	2	D300	25-MAY-26	1	107
3	3	D112	25-MAY-26	2	108
4	4	D406	31-MAY-26	2	108
5	5	D112	25-MAY-26	3	112
6	6	D100	25-JUL-26	3	109
7	7	D112	25-MAY-26	4	114
8	8	D300	25-MAY-26	4	111
9	9	D112	25-MAY-26	5	110

3. *Se vor majora cu 10% toate tarifele negociate din contractele semnate dupa 20-05-2026.*

```
UPDATE DETALII_CONTRACTE DC
SET DC.TARIF_NEGOCIAT = DC.TARIF_NEGOCIAT * 1.1
WHERE DC.ID_CONTRACT IN (SELECT C.ID_CONTRACT
                         FROM CONTRACTE C
                         WHERE C.DATA_SEMNARE > TO_DATE('20-05-2026', 'DD-MM-YYYY'));
```

	ID_CONTRACT	ID_SERVICIU	TARIF_NEGOCIAT
1	1	1	(null)
2	1	2	550
3	2	3	(null)
4	3	4	1125
5	4	7	1900
6	5	5	(null)
7	5	2	633
8	2	4	1100
9	2	5	450
10	3	5	(null)

14. Crearea unei vizualizări complexe

Vizualizare: detaliile contractelor și tariful negociat pentru fiecare serviciu.

```
CREATE OR REPLACE VIEW VIEW_SITUATIE_CONTRACTE AS
SELECT C.ID_CLIENT, C.NUME_FIRMA, CON.ID_CONTRACT, CON.DATA_SEMNARE,
DC.ID_SERVICIU, S.TARIF_LUNAR, DC.TARIF_NEGOCIAT
FROM CLIENTI C
JOIN CONTRACTE CON ON C.ID_CLIENT = CON.ID_CLIENT
JOIN DETALII_CONTRACTE DC ON CON.ID_CONTRACT = DC.ID_CONTRACT
JOIN SERVICII S ON S.ID_SERVICIU = DC.ID_SERVICIU;
```

proiect_bd.sql | VIEW_SITUATIE_CONTRACTE

Columns | Data | Grants | Dependencies | Details | Triggers | SQL | Errors

Sort.. | Filter:

	ID_CLIENT	NUME_FIRMA	ID_CONTRACT	DATA_SEMNARE	ID_SERVICIU	TARIF_LUNAR	TARIF_NEGOCIAT
1	1	NOVA PROMPT SRL	1	01-MAY-26	1	700	(null)
2	1	NOVA PROMPT SRL	1	01-MAY-26	2	600	550
3	2	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	3	950	(null)
4	2	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	4	1200	1100
5	2	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	5	450	450
6	2	ARGENTINE BAR & RESTAURANT SRL	3	12-MAY-26	4	1200	1125
7	2	ARGENTINE BAR & RESTAURANT SRL	3	12-MAY-26	5	450	(null)
8	3	KRONOS CONSULTING SRL	4	13-MAY-26	7	2000	1900
9	5	DELTA AGRO TRADING SRL	5	21-MAY-26	5	450	(null)
10	5	DELTA AGRO TRADING SRL	5	21-MAY-26	2	600	575

LMD permis: UPDATE

```
UPDATE V_SITUATIE_CONTRACTE
SET TARIF_NEGOCIAT = TARIF_NEGOCIAT * 1.05
WHERE ID_CONTRACT = 10 AND ID_SERVICIU = 3;
```

LMD nepermis: INSERT (un INSERT care încearcă să adauge date în mai multe tabele fizice simultan prin intermediul vizualizării este strict interzis)

```
INSERT INTO VIEW_SITUATIE_CONTRACTE (ID_CLIENT, NUME_FIRMA, ID_CONTRACT,
TARIF_NEGOCIAT)
VALUES (99, 'FIRMA NOUA SRL', 500, 1500);
```

Ivan Luca-Cristian
Grupa 141

15. Formulați în limbaj natural și implementați în SQL 3 cereri care să conțină: outer join pe minimum 4 tabele, division și analiză top-n

1. *Să se afișeze o listă cu toți clienții din baza de date: numele firmelor, numărul contractelor semnate, denumirea serviciilor incluse în acele contracte și descrierea pontajelor efectuate. Raportul include absolut toți clienții (chiar dacă nu au contracte), toate contractele (chiar dacă nu au servicii asociate încă) și toate serviciile din contracte (chiar dacă nu s-a lucrat/pontat nimic pentru ele în luna curentă). (outer join)*

```
SELECT C.NUME_FIRMA, CO.ID_CONTRACT, CO.DATA_SEMNARE, S.DENUMIRE AS DENUMIRE_SERVICIU, P.DATA_PONTAJ, P.NR_ORE
FROM CLIENTI C
LEFT OUTER JOIN CONTRACTE CO ON C.ID_CLIENT = CO.ID_CLIENT
LEFT OUTER JOIN DETALII_CONTRACTE DC ON CO.ID_CONTRACT = DC.ID_CONTRACT
LEFT OUTER JOIN SERVICII S ON DC.ID_SERVICIU = S.ID_SERVICIU
LEFT OUTER JOIN PONTAJE P ON (P.ID_CLIENT = C.ID_CLIENT AND P.ID_SERVICIU = S.ID_SERVICIU)
ORDER BY C.NUME_FIRMA, CO.ID_CONTRACT;
```

```
SELECT C.NUME_FIRMA, CO.ID_CONTRACT, CO.DATA_SEMNARE, S.DENUMIRE AS DENUMIRE_SERVICIU, P.DATA_PONTAJ, P.NR_ORE
FROM CLIENTI C
LEFT OUTER JOIN CONTRACTE CO ON C.ID_CLIENT = CO.ID_CLIENT
LEFT OUTER JOIN DETALII_CONTRACTE DC ON CO.ID_CONTRACT = DC.ID_CONTRACT
LEFT OUTER JOIN SERVICII S ON DC.ID_SERVICIU = S.ID_SERVICIU
LEFT OUTER JOIN PONTAJE P ON (P.ID_CLIENT = C.ID_CLIENT AND P.ID_SERVICIU = S.ID_SERVICIU)
ORDER BY C.NUME_FIRMA, CO.ID_CONTRACT;
```

Script Output | Query Result

SQL | All Rows Fetched: 20 in 0.019 seconds

	NUME_FIRMA	ID_CONTRACT	DATA_SEMNARE	DENUMIRE_SERVICIU	DATA_PONTAJ	NR_ORE
1	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	Consultanta Financiara Permanenta	20-MAY-26	7
2	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	Consultanta Financiara Permanenta	21-MAY-26	6
3	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	Suport si Monitorizare SAF-T	23-MAY-26	8
4	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	Contabilitate de Gestiune si Management	(null)	(null)
5	ARGENTINE BAR & RESTAURANT SRL	2	03-MAY-26	Consultanta Financiara Permanenta	23-MAY-26	8
6	ARGENTINE BAR & RESTAURANT SRL	3	12-MAY-26	Suport si Monitorizare SAF-T	23-MAY-26	8
7	ARGENTINE BAR & RESTAURANT SRL	3	12-MAY-26	Contabilitate de Gestiune si Management	(null)	(null)
8	BRAVE IT SOFTWARE SRL	8	22-MAY-26	(null)	(null)	(null)
9	DELTA AGRO TRADING SRL	5	21-MAY-26	Suport si Monitorizare SAF-T	(null)	(null)
10	DELTA AGRO TRADING SRL	5	21-MAY-26	Salarizare si Gestiune HR	23-MAY-26	4
11	DELTA AGRO TRADING SRL	5	21-MAY-26	Salarizare si Gestiune HR	22-MAY-26	6
12	KRONOS CONSULTING SRL	4	13-MAY-26	Pachet Corporate (Full-Service)	(null)	(null)
13	NEXUS LOGISTIC TRANSPORTER SRL	(null)	(null)	(null)	(null)	(null)
14	NOVA PROMPT SRL	1	01-MAY-26	Salarizare si Gestiune HR	(null)	(null)
15	NOVA PROMPT SRL	1	01-MAY-26	Contabilitate Financiara	23-MAY-26	8
16	PRIME DISTRIBUTION NET SRL	10	16-MAY-26	(null)	(null)	(null)
17	SMART RETAIL CONCEPT SRL	(null)	(null)	(null)	(null)	(null)
18	VIP MEDICA PREST SRL	6	21-MAY-26	(null)	(null)	(null)
19	VIP MEDICA PREST SRL	7	22-MAY-26	(null)	(null)	(null)
20	ZETA CONSTRUCT ENGINEERING SRL	9	08-MAY-26	(null)	(null)	(null)

2. *Să se afișeze ID-ul, numele și prenumele angajaților care sunt atât de activi încât au efectuat pontaje pentru absolut toți clienții înregistrați în baza de date a firmei. (division)*

```
SELECT A.ID_ANGAJAT, A.NUME, A.PRENUME
FROM ANGAJATI A
WHERE NOT EXISTS (SELECT C.ID_CLIENT
            FROM CLIENTI C
            WHERE NOT EXISTS (SELECT DATA_PONTAJ
                        FROM PONTAJE P
                        WHERE P.ID_ANGAJAT = A.ID_ANGAJAT AND P.ID_CLIENT =
C.ID_CLIENT));
```

```
SELECT A.ID_ANGAJAT, A.NUME, A.PRENUME
FROM ANGAJATI A
WHERE NOT EXISTS (SELECT C.ID_CLIENT
                  FROM CLIENTI C
                  WHERE NOT EXISTS (SELECT DATA_PONTAJ
                                    FROM PONTAJE P
                                    WHERE P.ID_ANGAJAT = A.ID_ANGAJAT AND P.ID_CLIENT = C.ID_CLIENT));
```

Script Output | Query Result

SQL | All Rows Fetched: 0 in 0.013 seconds

ID_AN...	NUME	PRENU...

(din pacate nu avem incă angajați atât de muncitori...)

3. *Să se determine topul primilor 3 angajați care au efectuat cele mai multe ore de muncă documentate în tabela de pontaje, afișând ID-ul angajatului, numele acestuia și numărul total de ore lucrate. (top-n)*

```
SELECT ROWNUM AS POZITIE, ID_ANGAJAT, NUME, PRENUME, TOTAL_ORE_LUCRATE
FROM (SELECT A.ID_ANGAJAT, A.NUME, A.PRENUME, SUM(P.NR_ORE) AS
TOTAL_ORE_LUCRATE
    FROM ANGAJATI A
    JOIN PONTAJE P ON A.ID_ANGAJAT = P.ID_ANGAJAT
    GROUP BY A.ID_ANGAJAT, A.NUME, A.PRENUME
    ORDER BY TOTAL_ORE_LUCRATE DESC)
WHERE ROWNUM <= 3;
```

Ivan Luca-Cristian
Grupa 141

```sql
-- analiza top-n
SELECT ROWNUM AS POZITIE, ID_ANGAJAT, NUME, PRENUME, TOTAL_ORE_LUCRATE
FROM (SELECT A.ID_ANGAJAT, A.NUME, A.PRENUME, SUM(P.NR_ORE) AS TOTAL_ORE_LUCRATE
      FROM ANGAJATI A
      JOIN PONTAJE P ON A.ID_ANGAJAT = P.ID_ANGAJAT
      GROUP BY A.ID_ANGAJAT, A.NUME, A.PRENUME
      ORDER BY TOTAL_ORE_LUCRATE DESC)
WHERE ROWNUM <= 3;
```

Query Result

SQL | All Rows Fetched: 3 in 0.064 seconds

	POZITIE	ID_ANGAJAT	NUME	PRENUME	TOTAL_ORE_LUCRATE
1	1	108	Pantic	Alberto	21
2	2	107	Hasnas	Calin	16
3	3	109	Mihalache	Andrei	10

16. Prezentarea planului de execuție a unei cereri complexe, optimizare/comparare plan alternativ folosind hint-uri si obiecte specifice optimizării cererilor

Alegem cererea 4 de la punctul 12 pentru optimizare.

```
SELECT C.ID_CLIENT, C.NUME_FIRMA,
       COUNT(CON.ID_CONTRACT) AS NUMAR_CONTRACTE,
       SUM(NVL(CALC_PLATI.TOTAL_ACHITAT, 0)) AS TOTAL_PLATIT_CLIENT
FROM CLIENTI C
LEFT JOIN CONTRACTE CON ON C.ID_CLIENT = CON.ID_CLIENT
LEFT JOIN FACTURI F ON CON.ID_CONTRACT = F.ID_CONTRACT
LEFT JOIN (SELECT ID_FACTURA, SUM(SUMA) AS TOTAL_ACHITAT
           FROM PLATI
           GROUP BY ID_FACTURA) CALC_PLATI ON F.ID_FACTURA = CALC_PLATI.ID_FACTURA
GROUP BY C.ID_CLIENT, C.NUME_FIRMA;
```

Script Output | Query Result | Explain Plan

SQL | 0.117 seconds

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			19	10
HASH		GROUP BY	19	10
HASH JOIN		OUTER	19	9
Access Predicates				
F.ID_FACTURA=CALC_PLATI.ID_FACTURA(+)				
HASH JOIN		OUTER	19	6
Access Predicates				
CON.ID_CONTRACT=F.ID_CONTRACT(+)				
HASH JOIN		OUTER	12	4
Access Predicates				
C.ID_CLIENT=CON.ID_CLIENT(+)				
TABLE ACCESS	CLIENTI	FULL	10	2
TABLE ACCESS	CONTRACTE	FULL	10	2
TABLE ACCESS	FACTURI	FULL	11	2
VIEW			1	3
HASH		GROUP BY	1	3
TABLE ACCESS	PLATI	FULL	1	2

[!] Se observă operația TABLE ACCESS FULL pe tabela PONTAJE, ceea ce indică o scanare ineficientă a întregului set de date.

Pentru o optimizare avansată, am creat un index compus de acoperire (Covering Index) pe coloanele (id_factura, suma). Scopul acestui index este de a stoca împreună atât cheia de legătură, cât și valorile financiare brute.

```
CREATE INDEX IDX_PLATI ON PLATI(ID_FACTURA, SUMA);
```

În noul plan de execuție, costul a scăzut substanțial. Optimizer-ul Oracle a realizat o optimizare de tip *Index-Only Access*. Deoarece indexul conține toate datele necesare subcererii, Oracle nu mai accesează deloc tabela fizică PLATI de pe disc, eliminând complet operațiile costisitoare de Input/Output asupra tabelului. (53 → 42)

Ivan Luca-Cristian
Grupa 141

```sql
SELECT C.ID_CLIENT, C.NUME_FIRMA,
       COUNT(CON.ID_CONTRACT) AS NUMAR_CONTRACTE,
       SUM(NVL(CALC_PLATI.TOTAL_ACHITAT, 0)) AS TOTAL_PLATIT_CLIENT
FROM CLIENTI C
LEFT JOIN CONTRACTE CON ON C.ID_CLIENT = CON.ID_CLIENT
LEFT JOIN FACTURI F ON CON.ID_CONTRACT = F.ID_CONTRACT
LEFT JOIN (SELECT ID_FACTURA, SUM(SUMA) AS TOTAL_ACHITAT
           FROM PLATI
           GROUP BY ID_FACTURA) CALC_PLATI ON F.ID_FACTURA = CALC_PLATI.ID_FACTURA
GROUP BY C.ID_CLIENT, C.NUME_FIRMA;

CREATE INDEX IDX_PLATI ON PLATI(ID_FACTURA, SUMA);
```

Script Output × | Query Result × | Explain Plan ×

SQL | 0.106 seconds

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			19	8
HASH		GROUP BY	19	8
HASH JOIN		OUTER	19	7
Access Predicates				
F.ID_FACTURA=CALC_PLATI.ID_FACTURA(+)				
HASH JOIN		OUTER	19	6
Access Predicates				
CON.ID_CONTRACT=F.ID_CONTRACT(+)				
HASH JOIN		OUTER	12	4
Access Predicates				
C.ID_CLIENT=CON.ID_CLIENT(+)				
TABLE ACCESS	CLIENTI	FULL	10	2
TABLE ACCESS	CONTRACTE	FULL	10	2
TABLE ACCESS	FACTURI	FULL	11	2
VIEW			1	1
HASH		GROUP BY	1	1
INDEX	IDX_PLATI	FULL SCAN	1	1

Ivan Luca-Cristian
Grupa 141

17. Realizarea normalizării BCNF, FN4, FN5. Aplicarea denormalizării, justificând necesitatea acesteia.

a) BCNF

Forma normală Boyce-Codd se bazează pe dependenţele funcţionale care iau în considerare toate cheile candidat dintr-o relaţie.

BCNF este mai strictă decât FN3. Diferența apare atunci când există chei candidate care se suprapun (au atribute comune): FN3 poate fi satisfăcută dacă partea dreaptă a dependenței este un atribut prim, în timp ce BCNF este violată dacă partea stângă nu este o supercheie.

Exemplu de încălcare BCNF:

Extindem schema bazei de date cu o relație care gestionează experții contabili alocați pentru reviziile fiscale ale clienților: AUDIT_FISCAL (id_client, tip_impozit, id_expert)

Regulile de business ale firmei:

Fiecare expert contabil este specializat și lucrează pe un singur tip de impozit (ex: doar TVA sau doar Impozit pe Profit): id_expert → tip_impozit

Dar la un anumit tip de impozit pot lucra mai mulți experți independenți.

Pentru un anumit client și un anumit tip de impozit, se alocă cel mult un singur expert expert contabil: (id_client, tip_impozit) → id_expert

Chei candidate:

CC1 = {id_client, tip_impozit} deoarece determină unicul expert alocat.

CC2 = {id_client, id_expert} deoarece id_expert → tip_impozit, deci perechea determină toate atributele.

Atribute prime (membre ale cel puțin unei chei candidate): id_client, tip_impozit, id_expert (toate atributele sunt prime).

Verificare FN3: Dependența id_expert→tip_impozit are în stânga un atribut care nu este supercheie, însă în dreapta are tip_impozit, care este un atribut prim (membru al CC1). Prin urmare, excepția din FN3 se aplică, iar relația este în FN3.

Verificare BCNF: În cadrul aceleiași dependențe id_expert → tip_impozit, determinatorul id_expert nu este supercheie. Din acest motiv, BCNF este încălcată.

Descompunere în BCNF:

Se elimină dependența problematică prin proiecție pe stânga ei, rezultând două relații distincte:

Relație	Atribute	Cheie primară
ALOCARE_EXPERT	id_client, id_expert	(id_client, id_expert)
SPECIALIZARE_EXPERT	id_expert, tip_impozit	id_expert

Descompunerea BCNF este lossless (fără pierderi de date la JOIN), însă nu preservează dependențele. Dependența originală (id_client, tip_impozit) → id_expert nu mai poate fi verificată local în nicio masă fără o operație de JOIN inter-tabele, un cost computațional asumat în BCNF.

b) FN4

FN4 elimină redundanțele datorate relațiilor m:n, adică datorate dependenței multiple. O relație este în a patra formă nomală dacă și numai dacă este în BCNF și nu conține relații m:n independente.

Exemplu de încălcare FN4:

Firma dorește să urmărească competențele tehnice și de portofoliu ale angajaților săi într-un tabel centralizat, reținând industriile clienților deserviți și aplicațiile software pe care aceștia le stăpânesc.

COMPETENTE_ANGAJAT(id_angajat, industrie_client, soft_contabil)

Regula de business:

Un angajat poate asigura servicii pentru mai multe industrii și poate utiliza mai multe aplicații software. Aceste două dimensiuni sunt complet independente între ele, dar ambele depind direct de angajat.

Dependențe multivalorice:

id_angajat ↠ industrie_client (un angajat are o mulțime de industrii alocate, indiferent de softurile folosite).

id_angajat ↠ soft_contabil (un angajat are o mulțime de softuri cunoscute, indiferent de industriile clienților).

Conform teoremei complementului, modificarea datelor introduce anomalii: dacă angajatul 501 învață un al treilea software, va trebui să inserăm manual câte o linie nouă pentru fiecare industrie din portofoliul său, generând o redundanță masivă a datelor.

Verificare FN4: În cadrul dependenței id_angajat ↠ industrie_client, atributul determinator id_angajat singur nu este supercheie (cheia completă fiind formată din toate cele 3 atribute). Astfel, FN4 este încălcată.

Descompunere în FN4
Fiecare dependență multivalorică independentă este extrasă într-o relație separată:

Relație	Atribute	Cheie primară
PORTOFOLIU_INDUSTRIE	id_angajat, industrie_client	(id_angajat, industrie_client)
SOFT_COMPETENTA	id_angajat, soft_contabil	(id_angajat, soft_contabil)

Avantaj: Adăugarea unui nou software de către un angajat necesită acum inserarea unei singure linii în tabela SOFT_COMPETENTA, fără a mai afecta sau duplica înregistrările din PORTOFOLIU_INDUSTRIE. Reconstituirea setului complet de date se realizează prin JOIN pe baza coloanei id_angajat (lossless).

c) FN5

Exemplu de încălcare FN5:

Firma de contabilitate organizează activitatea pe bază de echipe, servicii și clienți, utilizând un tabel care centralizează repartizarea sarcinilor: REPARTIZARE_PROIECT(id_angajat, id_client, tip_serviciu)

Regula de business (constrângere ciclică):

Un angajat execută un serviciu specific pentru un anumit client dacă și numai dacă sunt îndeplinite simultan următoarele condiții corelate:

- Angajatul este alocat în echipa de lucru a acelui client: (id_angajat, id_client).
- Angajatul deține competența tehnică de a oferi acel serviciu: (id_angajat, tip_serviciu).
- Clientul are acel tip de serviciu stipulat în contractul său: (id_client, tip_serviciu).

Verificarea dependenței de reuniune:

(id_angajat, id_client)	(id_angajat, tip_serviciu)	(id_client, tip_serviciu)
(A1, C101)	(A1, Audit)	(C101, Audit)
(A2, C101)	(A1, Consultanță)	(C101, Consultanță)
(A3, C102)	(A2, Audit)	(C102, Audit)
	(A3, Audit)	(C102, Salarizare)
	(A3, Salarizare)	

Dacă executăm un JOIN triplu pe baza acestor trei proiecții binare intermediare, vom obține exact înregistrările din relația inițială, ceea ce înseamnă că dependența de reuniune este satisfăcută.

Verificare FN5: Cheia candidată a relației este întreaga structură formată din (id_angajat, id_client, tip_serviciu). Dependența de reuniune nu este generată de structura acestei chei, ci este impusă de constrângerea ciclică din logica de business. Prin urmare, FN5 este încălcată.

Problema concretă: Dacă angajatul A2 dobândește competența de a efectua „Consultanță", iar clientul C101 are deja contractat serviciul de „Consultanță", baza de date va genera automat o inconsistență semantică dacă utilizatorul nu introduce manual al treilea rând în tabel pentru a închide cercul regulii de business.

Descompunere în FN5

Pentru ca regula ciclică să fie integrată nativ în structura bazei de date, tabelul original se descompune în cele trei proiecții binare corespunzătoare:

Relație	Atribute	Cheie primară
ECHIPA_CLIENT	id_angajat, id_client	(id_angajat, id_client)
COMPETENTA_SERVICIU	id_angajat, tip_serviciu	(id_angajat, tip_serviciu)
CONTRACT_SERVICIU	id_client, tip_serviciu	(id_client, tip_serviciu)

Avantaj structural: Constrângerea ciclică se validează automat prin arhitectura tabelelor. Atunci când se introduce o nouă linie în COMPETENTA_SERVICIU pentru angajatul A2 (Consultanță), corelarea sa cu ECHIPA_CLIENT și CONTRACT_SERVICIU va genera automat tripletul valid la nivelul vederilor de raportare bazate pe JOIN, fără riscul apariției datelor lipsă sau incomplete.

d) De ce e utila denormalizarea?

Denormalizarea este un proces în care se combină două sau mai multe tabele normalizate într-o singură tabelă, pentru a îmbunătăți performanța și eficiența operațiilor de interogare asupra bazei de date. Acest proces implică reintroducerea redundanței controlate în baza de date.

În ciuda evidentelor dezavantaje, cum ar fi creșterea spațiului de stocare, denormalizarea are și avantaje:

- Performanță sporită: Prin combinarea tabelelor normalizate într-o singură tabelă, se poate reducenumărul de operații de îmbinare (join-uri) necesare pentru a obține datele. Acest lucru poate duce la o îmbunătățire semnificativă a performanței în cazul interogărilor complexe și a volumelor mari de date.
- Simplificarea interogărilor: Denormalizarea poate face interogările mai simple și mai ușor de scris, deoarece datele necesare sunt disponibile într-un singur loc. Nu este nevoie de join-uri complexe pentru a obține date din mai multe tabele.

- Reducerea complexității aplicației: Prin denormalizare, se poate reduce complexitatea aplicației, deoarece nu mai este nevoie de logica complexă de îmbinare a datelor din tabele separate.
- Optimizarea operațiilor de scriere: În anumite situații, denormalizarea poate îmbunătăți performanța operațiunilor de scriere, cum ar fi inserări, actualizări și ștergeri, deoarece nu este necesară actualizarea mai multor tabele.

Aici, abordarea denormalizării subliniază conceptul că, plasând toate datele într-un singur loc, ar putea elimina necesitatea căutării acelor fișiere multiple pentru a colecta aceste date. În cadrul bazei mele de date, luând spre exemplu relația COMPETENTE_ANGAJAT și descompunerea ei de la FN4, este inutil și mult mai costisitor din punct de vedere al timpului de executare să parcurgem datele separat și din PORTOFOLIU_INDUSTRIE și din SOFT_COMPETENTA printr-o operație suplimentară de JOIN atunci când avem nevoie de o imagine de ansamblu imediată asupra profilului unui contabil.

18. Exemplificarea isolation levels prin exemple de tranzacții care se execută în paralel în condiții de concurență, evidențiind efectele diferitelor niveluri de izolare asupra concurenței și integrității datelor

Nivelele de izolare în SQL sunt mecanisme care controlează modul în care tranzacțiile accesează și modifică datele simultan într-o bază de date multi-utilizator. Aceste nivele asigură consistența și integritatea datelor în timpul tranzacțiilor și previn anomalii precum dirty reads, non repeatable reads și phantoms.

O scurtă prezentare a nivelelor de izolare din Oracle (acesta nu are REPEATABLE READ și READ UNCOMMITED) și a anomaliilor asociate:

- READ COMMITTED: Acest nivel permite o vizibilitate a modificărilor numai după ce acestea au fost confirmate (comise). Datele necomise de alte tranzacții nu sunt vizibile. Cu toate acestea, o tranzacție poate experimenta non-repeatable reads, în care aceeași interogare returnează rezultate diferite în timpul execuției, din cauza modificărilor comise de alte tranzacții între citiri.

```
-- READ COMMITED --

--se creeaza datele de test
CREATE TABLE CLIENTI_EXP AS SELECT * FROM CLIENTI;
CREATE TABLE PLATI_EXP AS SELECT * FROM PLATI;

-- se seteaza nivelul de izolare implicit
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

-- se modifica id-ul unei plati
UPDATE PLATI_EXP
SET ID_PLATA = 99
WHERE ID_PLATA = 2;

-- se verifica ca apare in tabel
SELECT * FROM PLATI_EXP WHERE ID_PLATA = 99;

-- se da rollback
ROLLBACK;

-- se observa ca id_plata a revenit la valoarea ei originala
SELECT * FROM PLATI_EXP;
```

- SERIALIZABLE: La acest nivel, tranzacțiile sunt executate într-o ordine ‘serială’, ceea ce înseamnă că tranzacțiile nu pot interacționa în mod concurent. Aceasta asigură cea mai mare izolare și evită non-repeatable reads și phantom. Anomalia phantom apare atunci când o tranzacție observă înregistrări noi care apar între citiri, ca urmare a inserării sau ștergerii efectuate de alte tranzacții.

```
-- SERIALIZABILE --

-- este necesara conectarea de pe un utilizator diferit de admin

-- se seteaza nivelul de izolare
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;

-- se sterge un client specific
DELETE FROM CLIENTI_EXP
WHERE NUME_FIRMA = 'NOVA PROMPT SRL';

-- se creeaza un savepoint
SAVEPOINT salvare1;

-- se verifica faptul ca firma a disparut din vizualizare
SELECT * FROM CLIENTI_EXP;

-- se sterge un alt client
DELETE FROM CLIENTI_EXP
WHERE NUME_FIRMA = 'KRONOS CONSULTING SRL';

-- se creeaza al doilea savepoint
SAVEPOINT salvare2;

-- ambele firme au disparut
SELECT * FROM CLIENTI_EXP;

-- dam rollback la primul savepoint
ROLLBACK TO SAVEPOINT salvare1;

-- a doua firma stearsa apare, dar prima este stearsa definitiv
SELECT * FROM CLIENTI_EXP;

COMMIT;
```

19. Justificarea necesității/utilității migrării la o bază de date de tip NoSql

În faza inițială a proiectului, am utilizat o bază de date relațională (Oracle) potrivită pentru tranzacții financiare fixe (plăți, facturi) datorită suportului nativ pentru proprietățile ACID (Atomicitate, Consistență, Izolare, Durabilitate). Totuși, pe măsură ce firma de contabilitate se dezvoltă, apar volume mari de date nestructurate sau semi-structurate (extrase bancare JSON, fișiere XML de la ANAF, jurnale de activitate ale utilizatorilor).

De ce este utilă migrarea la NoSQL (Document-based)?

- **Schemă flexibilă (Schema-less):** În Oracle, adăugarea unei coloane noi necesită ALTER TABLE și oprirea temporară a aplicației. În MongoDB, documentele din aceeași colecție pot avea structuri diferite. Dacă un client nou din UE are nevoie de câmpuri speciale pentru declarații intracomunitare, le adăugăm doar în documentul lui, fără a afecta restul bazei de date.
- **Fără JOIN:** Analizând formele normale avansate (FN4, FN5), am observat că descompunerea excesivă împarte datele în zeci de tabele. Pentru a reconstitui profilul unui client, Oracle trebuie să facă multe operații costisitoare de JOIN. În NoSQL, stocăm datele asociate (industrii, softuri, contracte) imbricate (embedded) direct în interiorul documentului Client. Citirea se face printr-o singură interogare extrem de rapidă.
- **Scalabilitate Orizontală:** Bazele de date relaționale scalează vertical (necesită un server din ce în ce mai scump). NoSQL scalează orizontal prin distribuirea datelor pe mai multe servere ieftine în cloud.

Scenarii în care NoSQL este mai avantajos în proiect:

- **Activity Logs:** Urmărirea fiecărei acțiuni efectuate de contabili în sistem (ce fișiere au deschis, când s-au logat). Datele cresc masiv și nu au nevoie de relații complexe.
- **Managementul documentelor fiscale scanate:** Stocarea metadatelor pentru facturi, bonuri și chitanțe recepționate prin API-uri externe, unde structura XML/JSON trimisă de serverele guvernamentale se poate schimba periodic.

În MongoDB, datele nu se mai stochează în tabele și rânduri, ci în Colecții (Collections) și Documente (Documents) în format BSON (JSON binar).

Pentru a elimina JOIN-urile costisitoare dintre tabelele CLIENTI, CONTRACTE și FACTURI, am migrat datele într-o singură colecție centralizată numită clienti. Relațiile de tip 1:M (un client are mai multe contracte, un contract are mai multe facturi) sunt reprezentate direct prin imbricare (embedding) în interiorul aceluiași document.

Ivan Luca-Cristian
Grupa 141

Comenzile pentru crearea bazei de date (Colecțiilor)

În MongoDB, crearea bazei de date și a colecțiilor se face automat în momentul în care inserăm primul document. Totuși, putem folosi comenzi explicite în linia de comandă (Mongosh) pentru a le defini organizat:

```
// 1. Crearea sau comutarea pe baza de date a proiectului
use firma_contabilitate;

// 2. Crearea explicită a colecției pentru clienți
db.createCollection("clienti");

// 3. Crearea unei colecții pentru istoricul de acțiuni (Logs)
db.createCollection("loguri_sistem");
```

Comenzi pentru Inserare, Modificare și Ștergere

INSERARE

```
db.clienti.insertOne({
  "id_client": 1,
  "nume_firma": "NOVA PROMPT SRL",
  "cui": "15366582",
  "adresa": "Aleea Mizil 60, 032345 Bucuresti",
  "contracte": [
    {
      "id_contract": 1,
      "nr_contract": "CTR-2026-1",
      "data_semnare": ISODate("2026-05-01T00:00:00Z"),
      "facturi": [
        {
          "id_factura": 1,
          "serie_numar": "INV-2026-001",
          "data_emitere": ISODate("2026-05-02T00:00:00Z"),
          "valoare": 1300,
          "plati": [] // nicio plata inregistrata momentan
        }
      ]
    }
  ]
});
```

UPDATE

```
db.clienti.updateOne(
  { "id_client": 1 },
  {
    $set: { "adresa": "Bulevardul Unirii 10, 010801 Bucuresti" }
  }
);
```

DELETE

```
db.clienti.deleteOne({ "id_client": 1 });
```

Exemplificarea comenzilor pentru interogarea datelor, incluzând operațiuni de filtrare și sortare

Căutările în MongoDB folosesc metoda .find(). Primul parametru reprezintă criteriul de filtrare (clauza WHERE), iar al doilea parametru reprezintă proiecția (clauza SELECT).

Afișarea tuturor datelor pentru clientul care are CUI-ul '12409034':

```
db.clienti.find({ "cui": "12409034" });
```

Găsirea tuturor clienților care au cel puțin o factură emisă cu o valoare mai mare sau egală cu 2000 RON. *(Folosim operatorul punct (.) pentru a naviga în adâncimea documentului și operatorul $gte - Greater Than or Equal).*

```
db.clienti.find({
  "contracte.facturi.valoare": {$gte: 2000}
});
```

Extragerea listei cu numele firmelor și adresele acestora, eliminând afișarea vectorului masiv de contracte și a ID-ului intern. Rezultatul trebuie sortat alfabetic descrescător după numele firmei. *(În proiecție, 1 înseamnă că vrem să vedem câmpul, iar 0 înseamnă ascundere. În .sort(), 1 reprezintă sortare ascendentă, iar -1 reprezintă sortare descendentă)*

```
db.clienti.find(
  {},                                  // filtru gol = toti clientii
  { "nume_firma": 1, "adresa": 1, "_id": 0 }  // afiseaza numele si adresa
).sort(
  { "nume_firma": -1 });               // sortare desc
```

Ivan Luca-Cristian
Grupa 141

Găsirea clienților care au efectuat plăți utilizând modalitatea "Card bancar" în istoricul lor.

```
db.clienti.find({
  "contracte.facturi.plati.mod_plata": "Card bancar"
});
```