## **PROIECT BAZE DE DATE**

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

Studenții unei facultăți participă la stagii oferite de către companii. Stagiile pot fi de tip *internship* sau *training*. Stagiile de practică de tip *internship* au fiecare, o dată de început și o dată de final. Stagiile de practică de tip training au câte o dată de început și o dată de final pentru fiecare student admis. Opțional, participarea la unele de stagii de practică poate fi plătită de către companii.

Participarea la stagii de practică în anii terminali este supervizată de un cadru didactic și se finalizează cu acordarea unui singur calificativ (admis/respins), în funcție de adeverința completată de către companie.

Pentru a fi admis la un stagiu de practica de tip training este necesar ca un student să cunoască unele tehnologii, la un anumit nivel cerut de companie.

Pe perioada efectuării unui stagiu studenții pot fi coordonați de specialiști din compania respectivă.

În timpul unui stagiu de tip training compania desfășoară proiecte. Un student poate fi încadrat pe mai multe proiecte din cadrul trainingului, iar în fiecare dintre acestea poate lucra cu mai multe tehnologii (limbaje, sisteme). La finalul stagiului, pentru fiecare student, coordonatorii evaluează nivelul de cunoaștere a tehnologiilor cu care a lucrat studentul, acordând un calificativ pentru fiecare dintre acestea.

1. Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului.

Un cadru didactic poate superviza zero sau mai mulți studenți.

Fiecare student primește un calificativ admis sau respins.

O companie poate organiza unul sau mai multe stagii de practică.

Un student poate avea în cadrul stagiului, mai mulți coordonatori.

Un coordonator poate coordona în cadrul unui stagiu, mai mulți studenți.

etc.

1. Descrierea entităților, incluzând precizarea cheii primare.

|  |  |  |
| --- | --- | --- |
| **ENTITATE** | **CHEIE PRIMARA** | **OBSERVATII** |
| stagiu | stagiu\_id | Informații generale despre stagii propuse de companii |
| internship | stagiu\_id | Informații specifice despre stagii de tip internship, subentitate a entitatii stagiu |
| tehnologie | tehnologie\_id | Tehnologie se refera la limbaje de programare, cunostente de arhitectura sistemelor, framework-uri, baze de date etc. |
| specialist | specialist\_id | Specialiștii sunt persoane, angajați ai unei companii |

1. Descrierea relațiilor, incluzând precizarea cardinalității acestora.

|  |  |  |
| --- | --- | --- |
| **RELATIE** | **CARDINALITATE** | **OBSERVATII** |
| coordoneaza | specialist-(sudent, stagiu) one-to-many  student-(specialist,stagiu)  one-to-many  Stagiu-(speciaist, student)  one-to-many | Un student poate avea mai mulți coordonatori pe parcursul unui stagiu |
| participa | Student-stagiu many-to-many | Un student poate participa la mai multe stagii.  La un stagiu pot participa mai mulți studenți. |

etc.

1. Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor.

ENTITATE: SUPERVIZEAZA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atribut** | **Tip** | **Dimensiune/ precizie** | **Valori posibile si valori default** | **Observații, obligatoriu/opțional** |
| calificativ | string | 6 | admis, respins |  |

ENTITATE: STAGIU

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atribut** | **Tip** | **Dimensiune/ precizie** | **Valori posibile si valori default** | **Observații, obligatoriu/opțional** |
| data\_inceput | date |  |  |  |
| data\_final | date |  |  |  |
| Sesiune\_online\_prezentare | string |  |  | Link-uri către întâlniri online de prezentare a programului de stagiu.  Atribut multiplu |

RELATIE: PARTICIPA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atribut** | **Tip** | **Dimensiune/ precizie** | **Valori posibile si valori default** | **Observații, obligatoriu/opțional** |
| data\_inceput | date |  |  | Data la care un student începe stagiul, NOT NULL |
| data\_final | date |  |  | Data la care un student finalizează stagiul, |
| salariu | float | 6 | Valoare default 0 | Suma plătită de companie pentru participare la stagiu, NULL |

RELATIE: CUNOSTINTE\_NECESARE

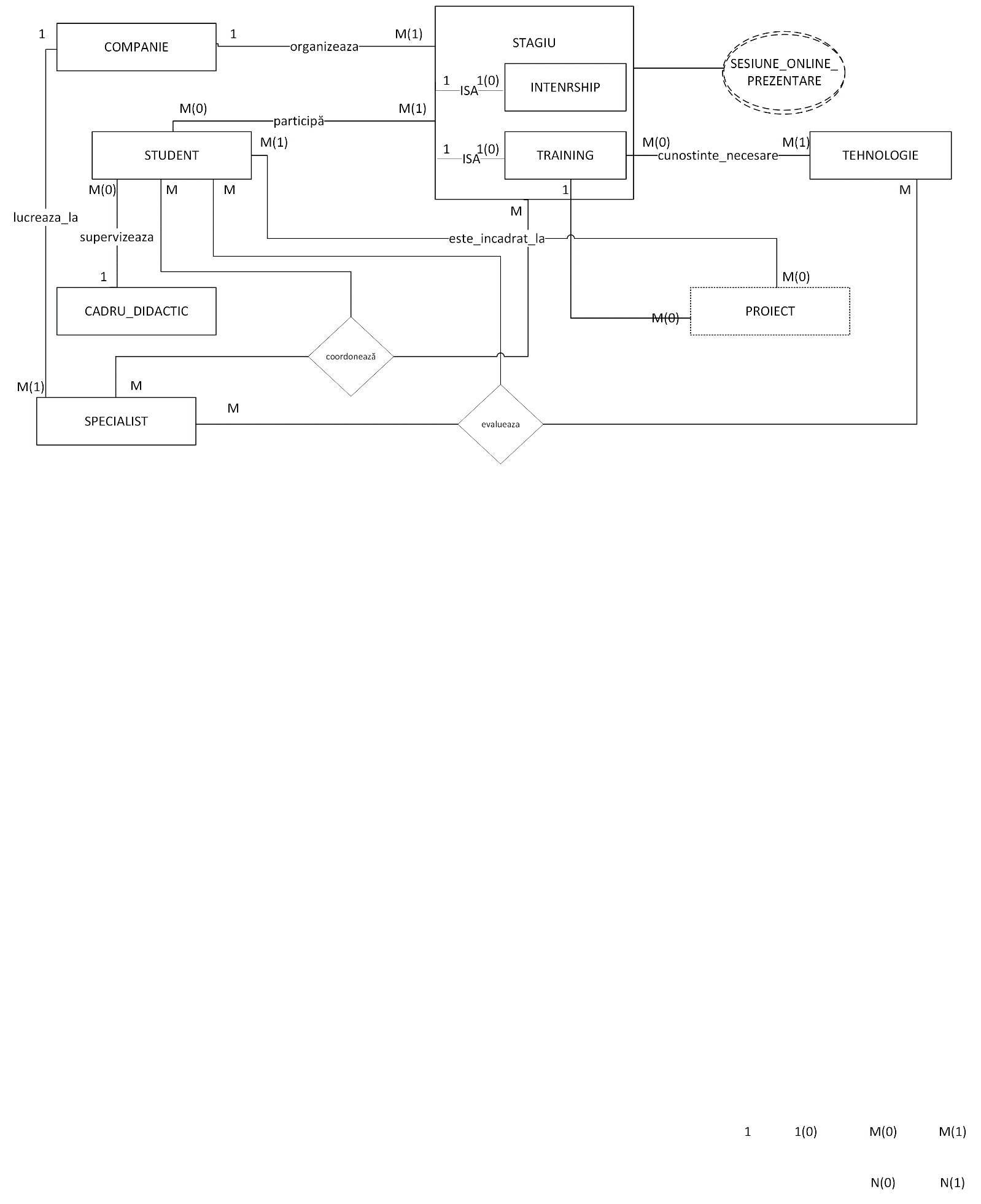
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atribut** | **Tip** | **Dimensiune/ precizie** | **Valori posibile si valori default** | **Observații, obligatoriu/opțional** |
| nivel | integer | 1 | [1,5] | Nivelul la care un candidat trebuie să cunoască o tehnologie pentru a fi admis, NOT NULL |

ENTITATE: SUPERVIZEAZA

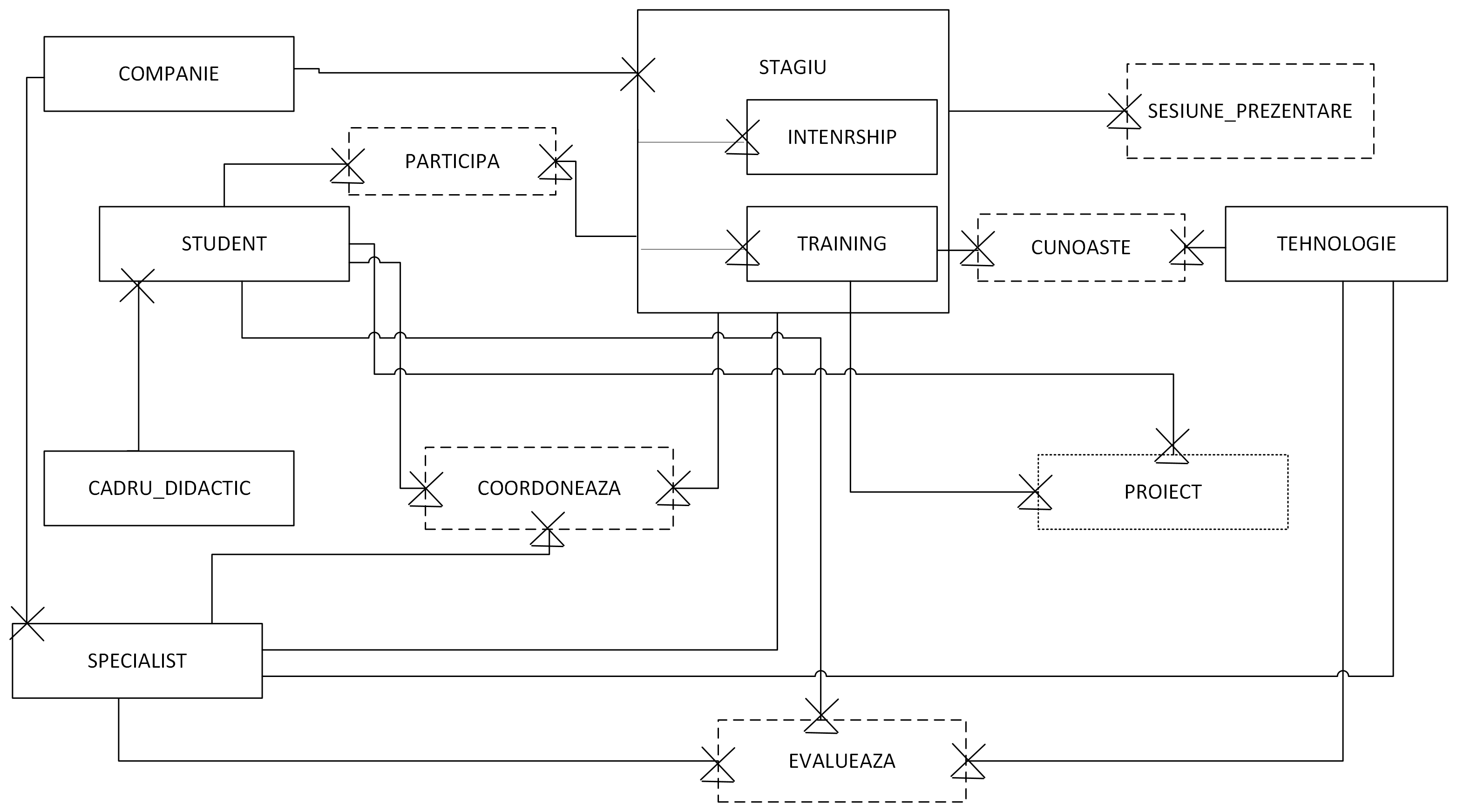
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atribut** | **Tip** | **Dimensiune/ precizie** | **Valori posibile si valori default** | **Observații, obligatoriu/opțional** |
| nivel | integer | 1 | [1,5] | Nivelul la care un candidat cunoaște o tehnologie, NOT NULL |

etc.

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



1. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6. Diagrama conceptuală obținută trebuie să conțină minimum 6 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ.



1. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.

COMPANIE(companie\_id#, denumire, adresa, site\_web)

STUDENT(student\_id#, nume, prenume, nr\_matricol, data\_nasterii)

COORDONEAZA(specialist\_id#, sudent\_id#, stagiu\_id#, feedback\_specialist)

EVALUEAZA(specialist\_id#, student\_id#, tehnologie\_id#, nivel, data\_evaluare)

etc.

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

Presupunem că tabelul corespunzător participării dezvoltatorilor la proiectele unei companii are schema relațională PARTICIPARE (cod\_student, nume\_student, cod\_proiect, denumire\_proiect, cod\_sef\_proiect, nume\_sef\_proiect, buget, data\_inceput\_proiect, data\_limita\_proiect, cod\_task, descriere\_task, data\_incepere\_task, durata\_realizare\_task).

1. Determinarea mulțimii dependențelor F funcționale care există între atributele acestei relații.

(cod\_student) (nume\_student)

(cod\_proiect) (denumire\_proiect)

(cod\_proiect) (cod\_sef\_proiect)

(cod\_sef\_proiect) (nume\_sef\_proiect)

(cod\_proiect) (buget)

(cod\_proiect) (data\_inceput\_proiect)

(cod\_proiect) (data\_limita\_proiect)

(cod\_task) (descriere\_task)

(cod\_task) (data\_incepere\_task)

(cod\_task) (cod\_proiect)

(cod\_student, cod\_task) (cod\_proiect, durata\_realizare\_task)

1. Aduceți relația la forma normală 3, justificând transformările care au loc la fiecare pas. (prin descompuneri fără pierderi de infamații, casey-delobel)

Normalizați PARTICIPARE în FN3 utilizând algoritmul de sinteză.

**Calculare forma canonica a lui *F*c:**

*F* = *F*c

Repetă

* Union rule: se înlocuiesc și cu
* Se elimină atribute din membrul stâng sau din membrul drept al unei dependențe:

Exemplu F = {} se poate elimina din

F = {}

Exemplu F = {} se poate elimina din

F = {}

Până când mulțimea F nu se modifică

**Algoritmul de sinteză**

Repetă

* Calculare Fc
* Pentru fiecare dependență din *F*c se adaugă o relație *Ri =*
* Dacă nicio relație *Ri* nu conține o cheie candidat, se adaugă o relație *Rk = K, K* cheie candidat pentru *R*
* Dacă există *i, j* astfel încât , se elimină

Până când mulțimea relațiilor { nu se modifică

c

* Union rule: se înlocuiesc și cu

(cod\_student) (nume\_student)

(cod\_proiect) (denumire\_proiect, cod\_sef\_proiect, buget, data\_inceput\_proiect, data\_limita\_proiect)

(cod\_sef\_proiect) (nume\_sef\_proiect)

(cod\_task) (descriere\_task, data\_incepere\_task, cod\_proiect)

* Se elimină atribute din membrul stâng sau din membrul drept al unei dependențe:

Exemplu F = {} se poate elimina din

F = {}

(cod\_student, cod\_task) (~~cod\_proiect~~, durata\_realizare\_task)

(cod\_task) (descriere\_task, data\_incepere\_task, cod\_proiect)

(cod\_student, cod\_task) ( durata\_realizare\_task)

Relații

R1(cod\_student, nume\_student)

R2(cod\_proiect, denumire\_proiect, cod\_sef\_proiect, buget, data\_inceput\_proiect, data\_limita\_proiect)

R3(cod\_sef\_proiect, nume\_sef\_proiect)

R4(cod\_task,descriere\_task, data\_incepere\_task, cod\_proiect)

* Dacă nicio relație *Ri* nu conține o cheie candidat, se adaugă o relație *Rk = K, K* cheie candidat pentru *R*

Daca nu avem R5(cod\_student, cod\_task, durata\_realizare\_task)

adaugam R5(cod\_student, cod\_task)

Relații in FN3.

R1(cod\_student, nume\_student)

R2(cod\_proiect, denumire\_proiect, cod\_sef\_proiect, buget, data\_inceput\_proiect, data\_limita\_proiect)

R3(cod\_sef\_proiect, nume\_sef\_proiect)

R4(cod\_task,descriere\_task, data\_incepere\_task, cod\_proiect)

R5(cod\_student, cod\_task, durata\_realizare\_task)