UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

BAZE DE DATE

Coordonator,

Prof. Vasile Silviu Laurentiu

Student,

Troian Ștefan

București, 2020

UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

ACTIVITATEA DE ÎNCHIRIERE A TROTINETELOR

ELECTRICE

Coordonator,

Prof. Vasile Silviu Laurentiu

Student,

Troian Ștefan

București, 2020

CUPRINS

1.PREZENTAREA MODELULUI ȘI A REGULILOR ACESTUIA...............................................3

2.DIAGRAMA ENTITATE-RELAȚIE.............................................................................................4

3.DESCRIEREA ENTITĂȚILOR, ATRIBUTELOR, CHEILOR, RELAȚIILOR ȘI A CARDINALITĂȚILOR.....................................................................................................................5

4.DIAGRAMA CONCEPTUALĂ.....................................................................................................6

5.DESCRIEREA CONSTRÂNGERILOR DE INTEGRITATE.......................................................7

6.SCHEMELE RELAȚIONALE (DESCRIEREA CONSTRÂNGERILOR ON DELETE CASCADE / SET NULL)................................................................................................................10

7.CREAREA TABELELOR (INCLUSIV A CONSTRÂNGERILOR)..........................................13

8.INTRODUCERE DATE...............................................................................................................17

1.PREZENTAREA MODELULUI ȘI A REGULILOR ACESTUIA

Segmentul transportului intraurban ce ține de trotinete, în special de trotinete electrice s-a dezvoltat enorm în ultimii ani, astfel că pentru unii prezenta lor a devenit una indispensabilă. Serviciile de închiriat trotinete electrice devin tot mai numeroase, iar avantajele sunt cât se poate de evidente precum deplasarea rapidă către locul dorit, faptul că nu poluează, petrecerea în aer liber , singur sau cu prietenii, într-un mod inedit și captivant etc.

Procedura de preluare a unui astfel de vehicul este destul de simplă:

-instalezi aplicatia companiei (disponibilă pentru iOS și Android)

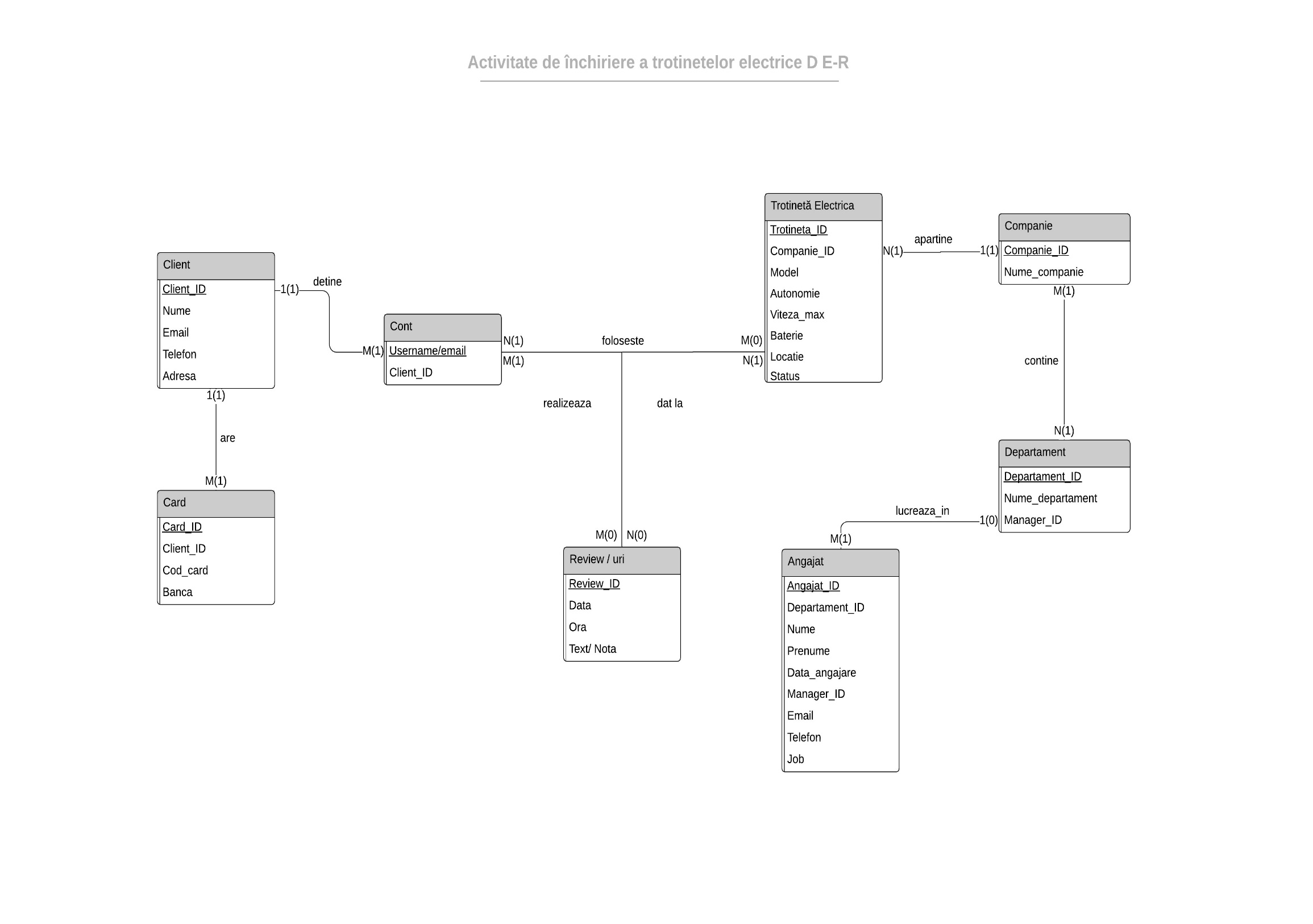
-introduci datele personale și datele cardului

-gasești o trotinetă electrică cu ajutorul hărtii interactive și îi scanezi codul QR

-cursa poate începe, iar odată ajunși la destinatie e de ajuns apăsarea butonului pentru încheierea cursei, iar pe ecran se va afisa instantaneu costul total al calatoriei și traseul parcurs.

Proiectul reprezintă un concept al bazei de date pentru ilustrarea activitatii de închiriere a trotinetelor electrice întrucat nu prezintă interfața disponibilă clientului, ci aceea de gestionare a bazei de date, dar și pentru incapacitățile de realizare a unei hărti interactive în timp real. Activitatea presupune înregistrarea unui client (tabelul Client) într-un cont (tabelul Cont) care poate deține mai multe carduri (tabelul Carduri). Prin intermediul contului acesta poate realiza o cursă (tabelul Cursă) cu o trotinetă (tabelul Trotinetă), fiind capabil sa poată da și un review (tabelul review) cursei. “In spate” există o companie (tabelul Companie) care deține trotinetele și care conține anumite departamente (tabelul departamente) în care lucrează angajați (tabelul Angajat).

2. DIAGRAMA ENTITATE-RELAȚIE.



3. DESCRIEREA ENTITĂȚILOR, ATRIBUTELOR, CHEILOR, RELAȚIILOR ȘI A CARDINALITĂȚILOR

Diagrama entitate-relație este formată din 8 tabele de sine stătătoare. Începând de la stânga la dreapta avem următoarele entitați și atribute:

* Tabelul **Client** are ca și cheie primara id-ul clientului. Restul atributelor sunt numele, email-ul, telefonul și adresa clientului.
* Tabelul **Card** conține id-ul cardului ca și cheie primară, id-ul clientului pentru a ști cui aparține cardul și datele cardului ilustrate prin atributele cod\_card și banca de care aparține.
* Tabelul **Cont** reprezintă modalitatea prin care un client poate închiria o trotinetă și conține un username, ce este unic și care reprezintă cheia primară, și id-ul clientului.
* Tabelul **Review** ilustrează posibilitatea clientului de a da un review unei curse. Acesta are ca și cheie primară id-ul review-ului, iar restul atributelor sunt data, ora și conținutul acestuia.
* Tabelul **Trotinetă\_electrică** conține date despre trotinetele utilizate și are cheia primara id-ul trotinetei. Celelalte atribute sunt id-ul companiei de care ține trotineta, modelul acesteia, autonomia (km), viteza maximă (km/h), bateria (%), locația la care se află și statusul acesteia (indisponibilă/disponibilă ca default).
* Tabelul **Companie** reține id-ul companiei, fiind și cheie primară, dar și numele acesteia care este unic.
* Tabelul **Departament** conține ca și cheie primară id-ul departamentului, restul atributelor fiind alcătuite din numele departamentului și manager-ul acestuia.
* Tabelul **Angajat** deține datele angajaților și are ca și cheie primară id-ul angajatului. Celelalte atribute sunt id-ul departamentului în care lucrează, numele și prenumele angajatului, data angajarii, managerul, email-ul , telefonul și denumirea jobului care îl prestează.

Relația dintre **Client** și **Card** este de tip one-to-many deoarece un client poate deține mai multe carduri. Cardinalitățile ilustrează faptul că un client poate avea mai multe carduri, dar minimum unul, iar un card este deținut doar de un client.

Între **Client** și **Cont** există o relație de tip one-to-many deoarece un client poate avea mai multe conturi. Prin intermediul cardinalităților întelegem că un client poate avea mai multe conturi, dar cel puțin unul, dar și că un cont aparține doar de un client și numai unul.

Relația dintre **Cont**, **Trotinetă\_electrică** și **Review** este una de tip 3, între ele existând un tabel asociativ în diagrama conceptuală. Relațiile între tabele sunt de tip many-to-many, iar cardinalitățile ilustrează că de pe un cont se pot realiza mai multe review-uri sau niciunul, dar și faptul că de pe el se pot închiria mai multe trotinete (la momente diferite) sau niciuna. Același review poate fi dat de pe mai multe conturi, dar neaparat unul, și tot acesta poate fi dat la minimum o trotinetă. O trotinetă poate fi închiriată de pe mai multe conturi, dar cel puțin unul și poate avea mai multe review-uri primite sau niciunul.

Relația **Companie-Trotineta\_electrica** este una one-to-many, iar cardinalitățile pun în evidentă faptul că o companie deține mai multe trotinete, dar cel puțin una, iar o trotinetă aparține neapărat doar de o companie.

Între **Companie** și **Departament** există o relație many-to-many, iar cardinalitățile ilustrează că o companie poate avea mai multe departamente, însă cel puțin unul, dar și faptul că un departament poate apărea în cadrul mai multor companii, dar neaparat în una.

Între **Departament** și **Angajat** este prezentă o relație de tipul one-to-many. Cardinalitățile subliniază ideea că un departament are mai mulți angajați, minimum unul, dar și că un angajat poate lucra într-un departament sau niciunul (președintele companiei).

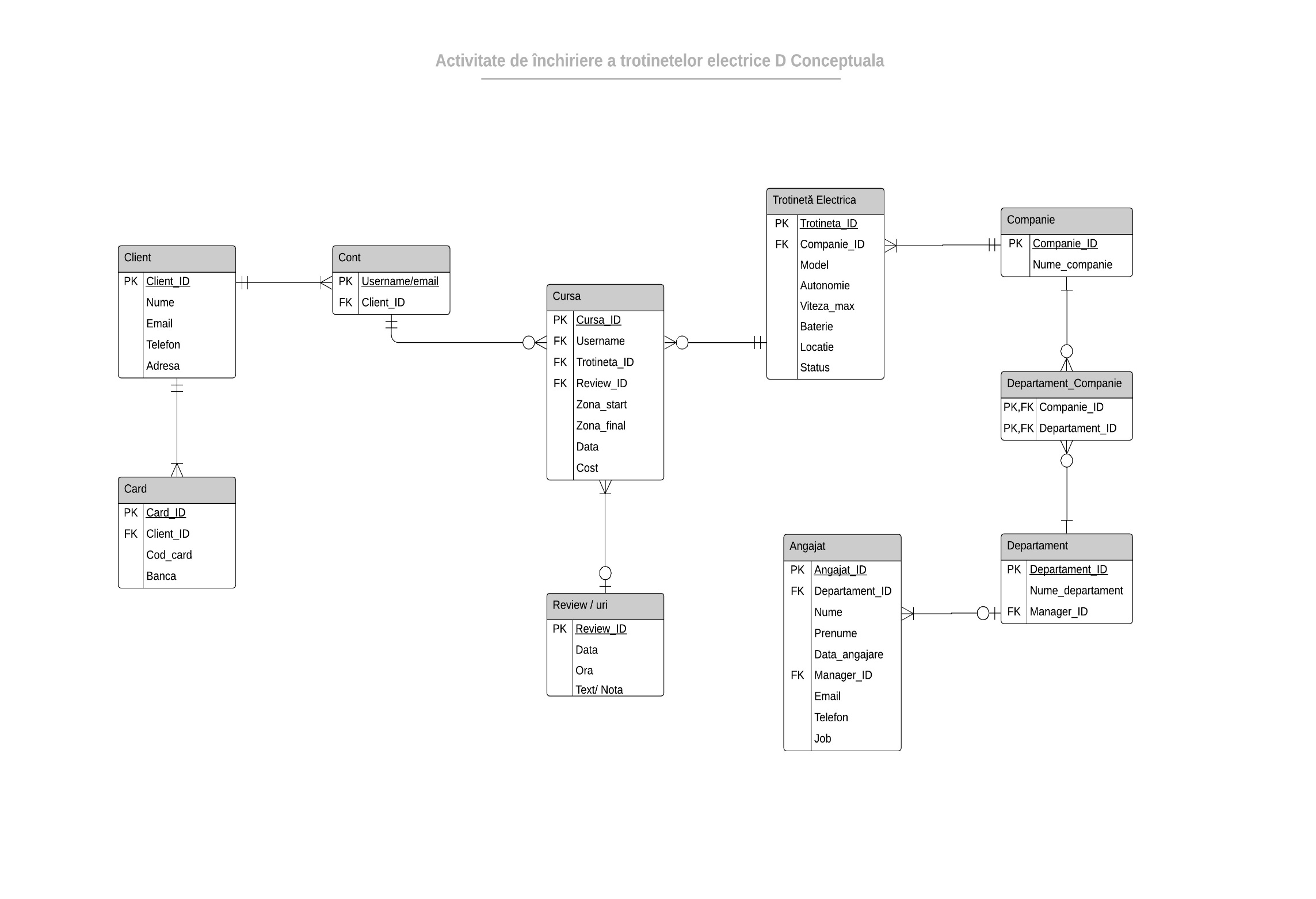
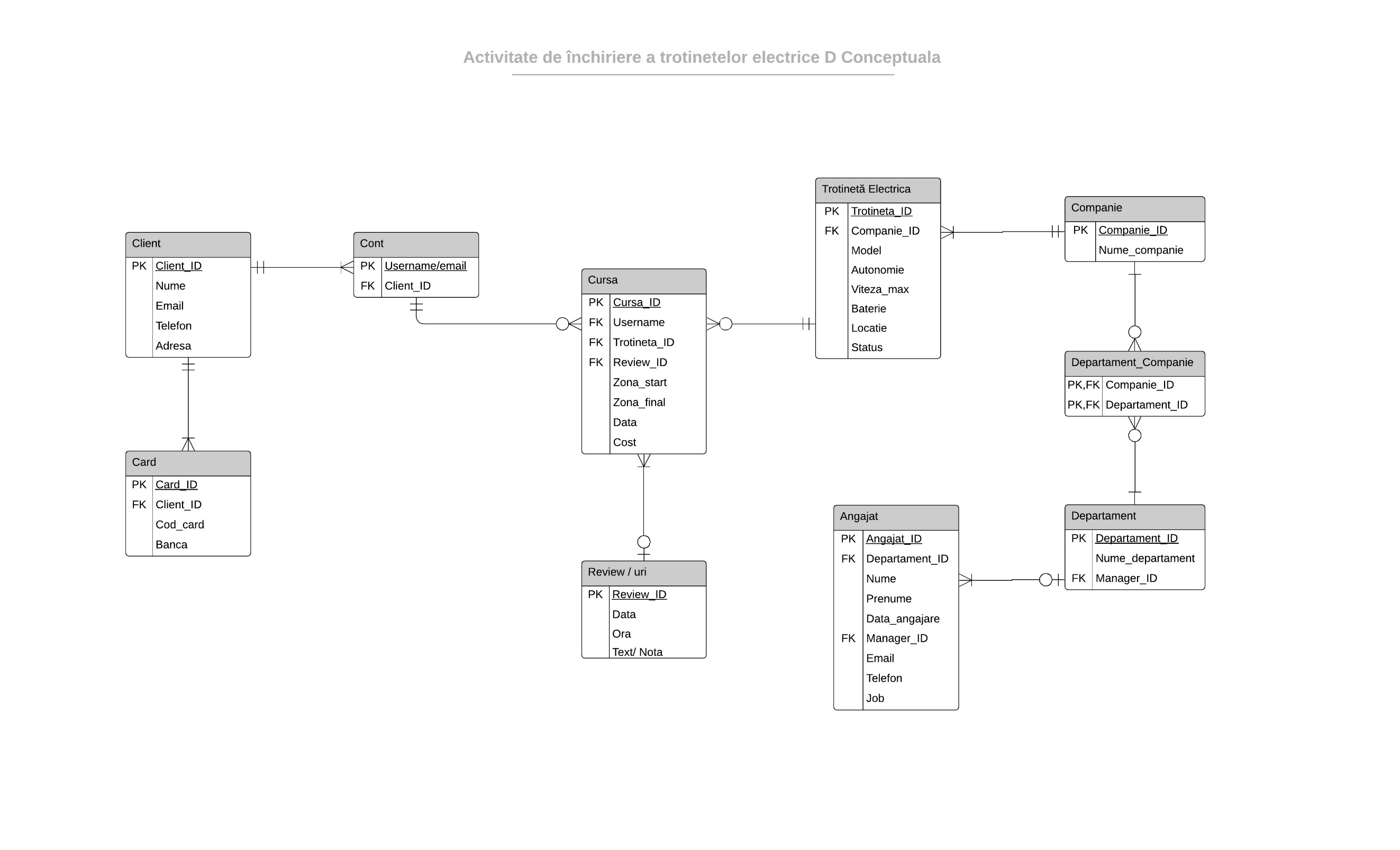
4.DIAGRAMA CONCEPTUALĂ

Diagrama conceptuală prezintă în plus față de cea entitate-relație 2 tabele asociative și anume tabelul **Cursă** și cel **Departament\_Companie**. Tabelul **Cursă** conține ca și cheie primară id-ul cursei, iar restul atributelor sunt username-ul dat de cont, id-ul trotinetei utilizate, id-ul review-ului, zona de start și de final, data când a fost realizată cursa și nu în ultimul rând costul acesteia. Tabelul **Departament\_Companie** conține atât ca și cheie primară și cheie externă id-urile companiei și departamentului, fiind un tabel asociativ ca urmare a unei legaturi many-to-many.

Relația dintre **Cont** și **Cursă** este one-to-many, de pe un cont având posibilitatea de a realiza mai multe curse, iar o cursă trebuie realizată neapărat de pe doar un cont. Între **Trotinetă\_electrică** și **Cursă** există o relație one-to-many deoarece o trotinetă poate apărea în mai multe curse, iar o cursă folosește doar o trotinetă. Relația dintre **Review** și **Cursă** este una tot de tip one-to-many întrucât același review poate fi dat la mai multe curse, iar o cursă poate sau nu avea un review.

Relațiile dintre **Departament\_Companie** și **Companie**, respectiv **Departament**, sunt de tip one-to-many deoarece o companie poate avea mai multe departamente, iar un departament se poate regăsi în cadrul mai multor companii.

5. DESCRIEREA CONSTRÂNGERILOR DE INTEGRITATE



Tabelul **Client** ca și constrângeri prezintă:

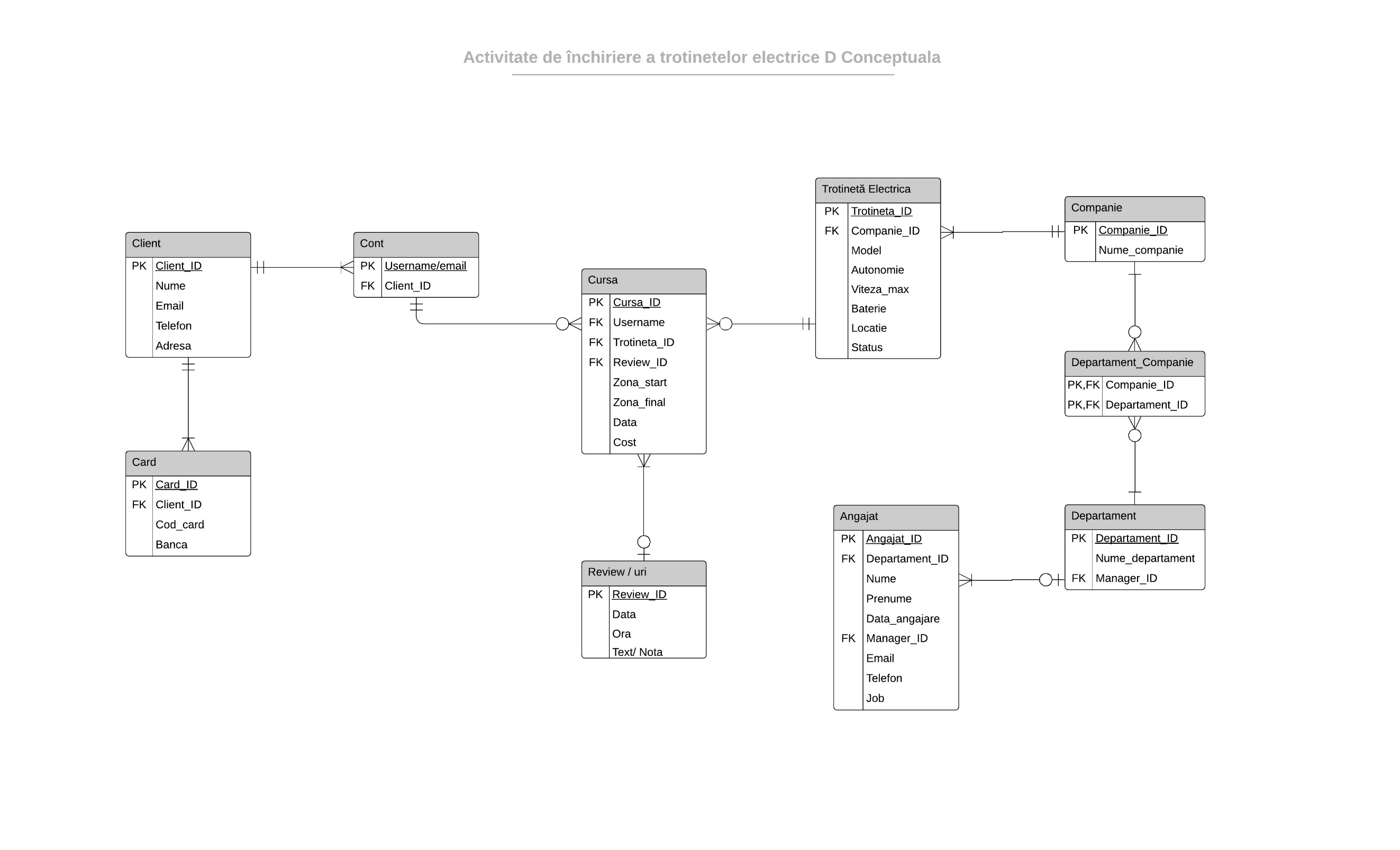
-Client\_id primary key, check > 0

-Nume not null

-Email not null

-Telefon not null

Client id primește primary key deoarece este atributul principal în tabel, iar check > 0 pentru inserarea doar dacă id-ul este pozitiv. Nume, Email și Telefon primesc constrângerea de tip not null întrucât e obligatoriu pentru un client să prezinte unele date personale.



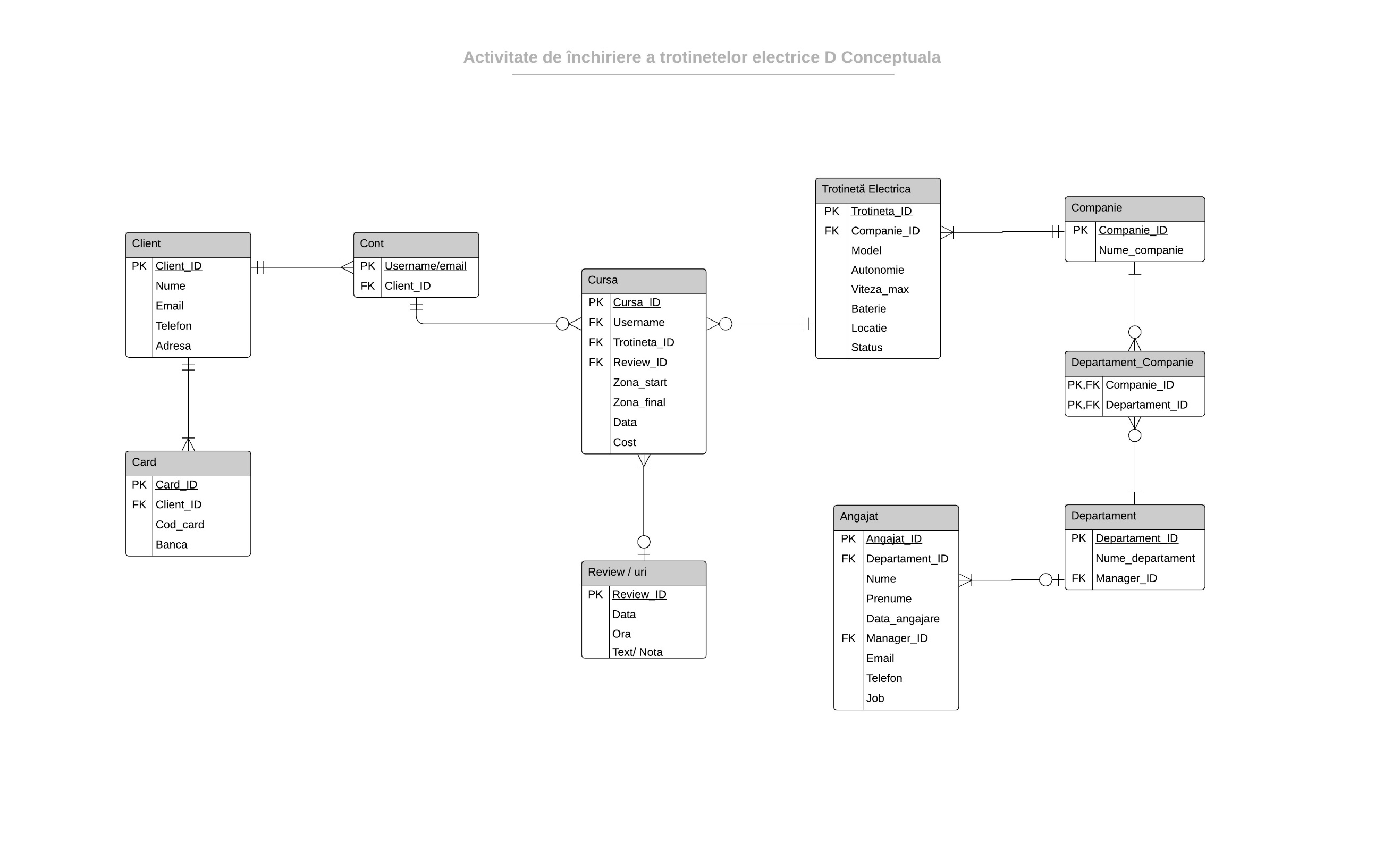
Tabelul **Card** conține urmatoarele constrângeri:

-Card\_id primary key, check > 0

-Client\_id foreign key + on delete cascade

-Cod\_card not null

Card\_id primește primary key și check > 0 pentru că reprezintă atributul principal din tabelul Card. Client\_id are ca și constrângere foreign key doarece este coloana de legatură cu un alt tabel, iar Cod\_card primeste not null pentru consistența datelor.

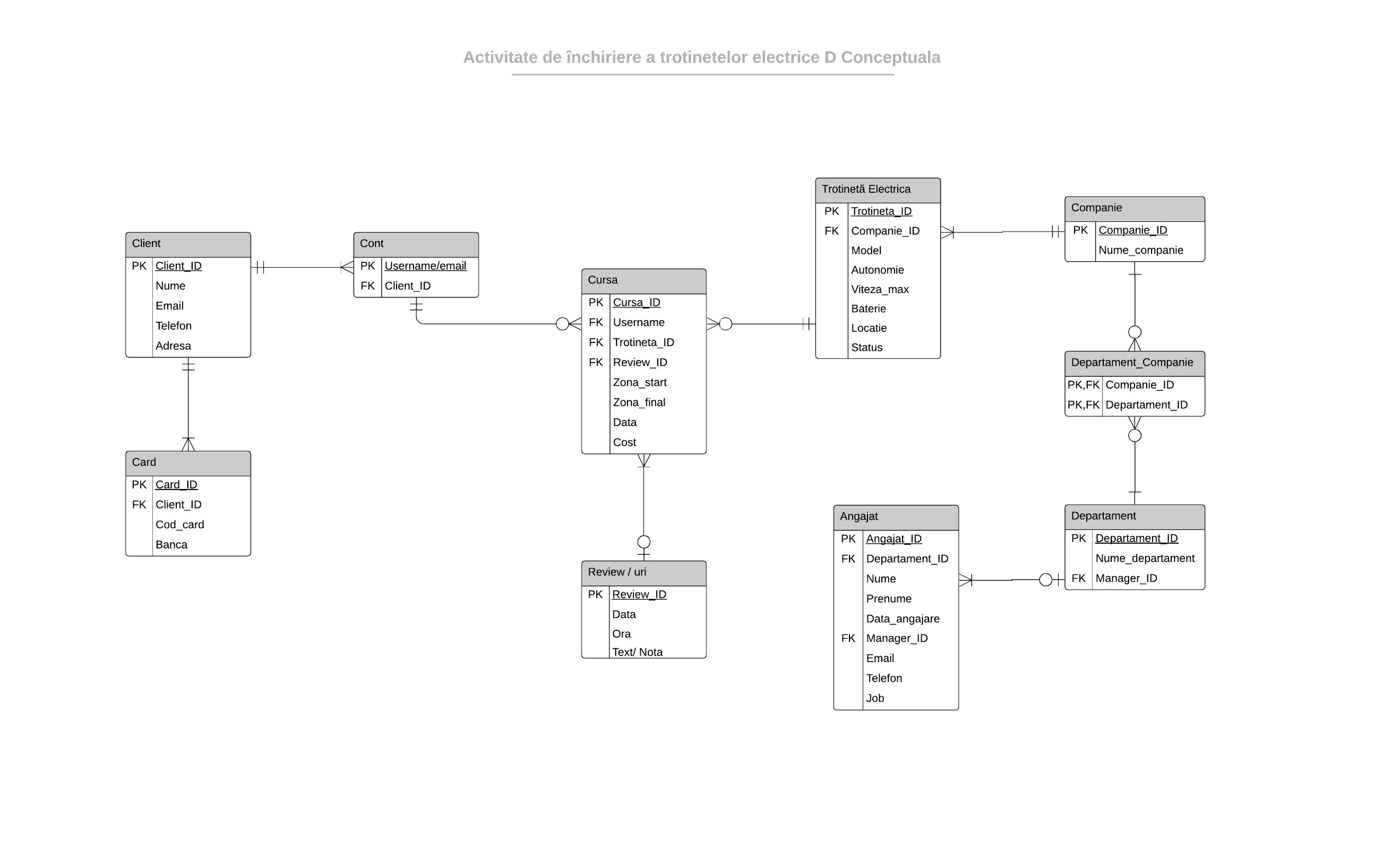


Tabelul **Cont** are urmatoarele constrângeri:

-Username primary key

-Client\_id foreign key + on delete cascade, not null

Tabelul are primary key pe atributul Username doarece este ilustrativ tabelului, iar foreign key pe Client\_id doarece este coloana de legatură cu un alt tabel și care trebuie să fie nenulă.

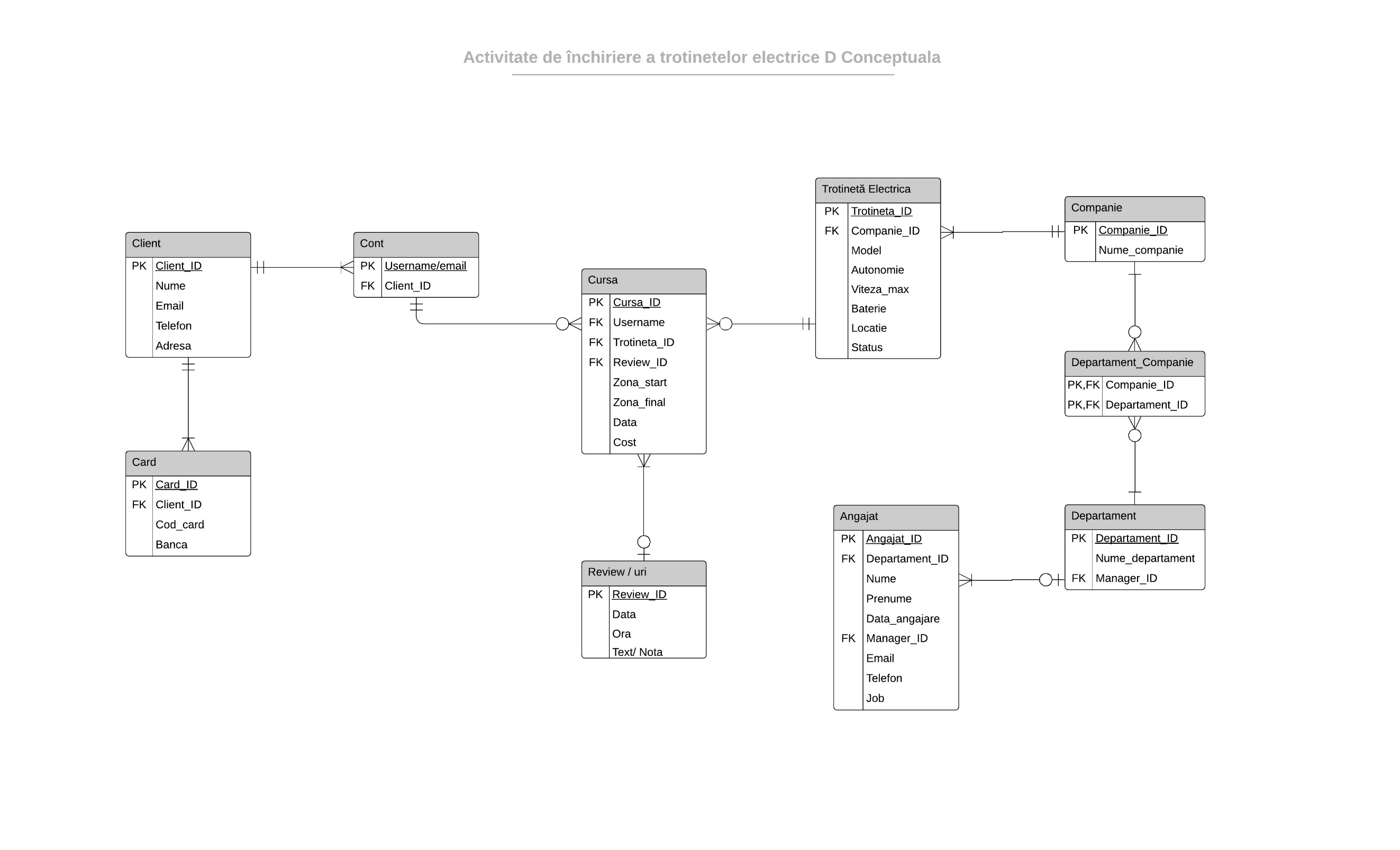
Tabelul **Review** conține la nivel de constrângeri:

-Review\_id primary key, check > 0

-Data default sysdate

-Ora not null

Atributul Review\_id primiește primary key doarece este reprezentativ pentru tabel, având necesitatea de a fi pozitiv. Data primeste ca default data curentă, iar Ora primește constrângerea de not null pentru deosebirea review-urilor date în aceeași zi.

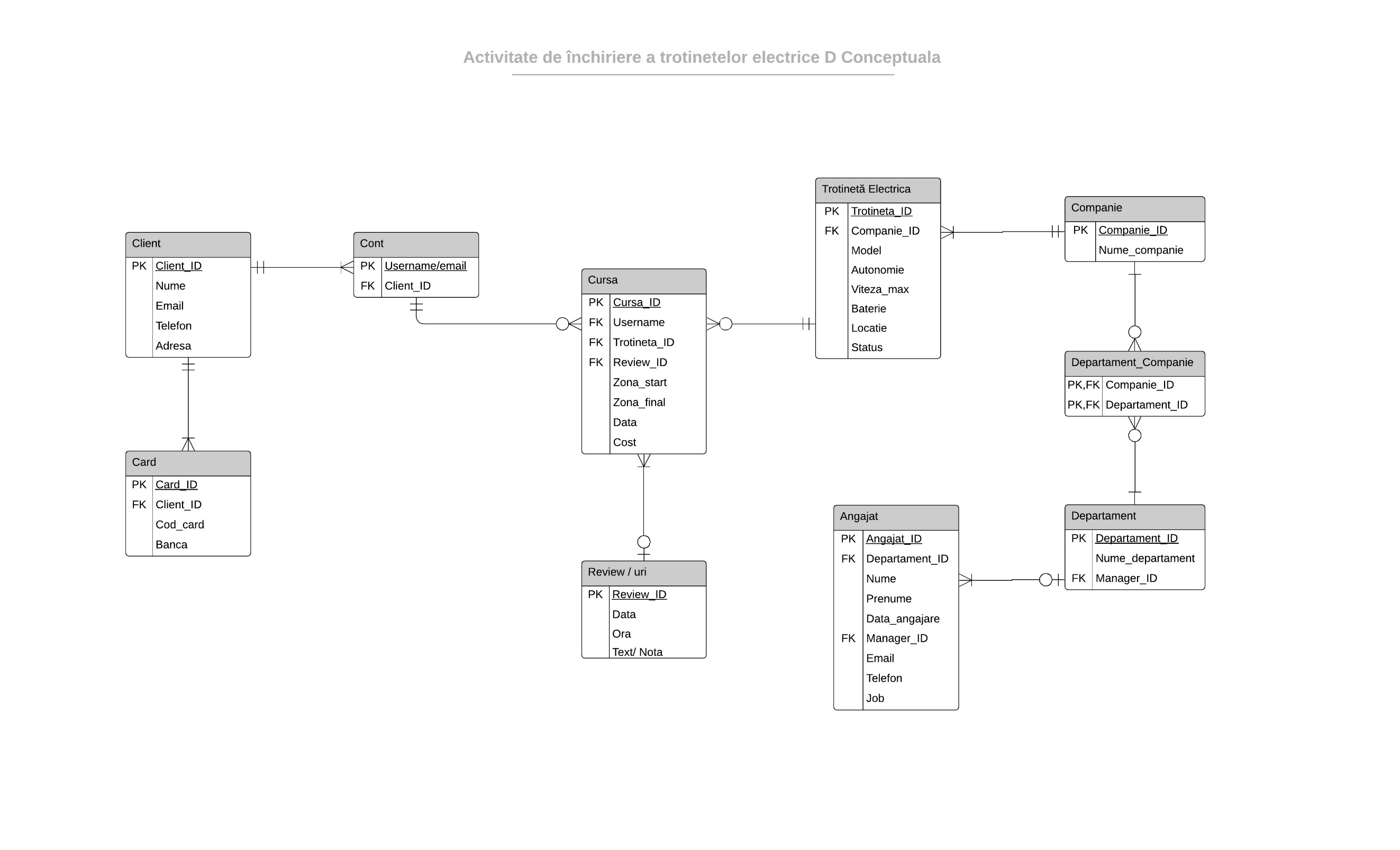


Tabelul **Companie** prezintă ca și constrângeri:

-Companie\_id primary key, check > 0

-Nume\_companie not null, unique

Tabelul are primary key pe companie\_id doarece atributul este ilustrativ pentru tabel și check > 0 pentru a avea un id pozitiv, iar not null și unique pentru inserarea obligatorie a unui nume unic pentru companie.

 Tabelul **Trotinetă\_electrică** are urmatoarele constrângeri:

-Trotineta\_id primary key, check-> 0

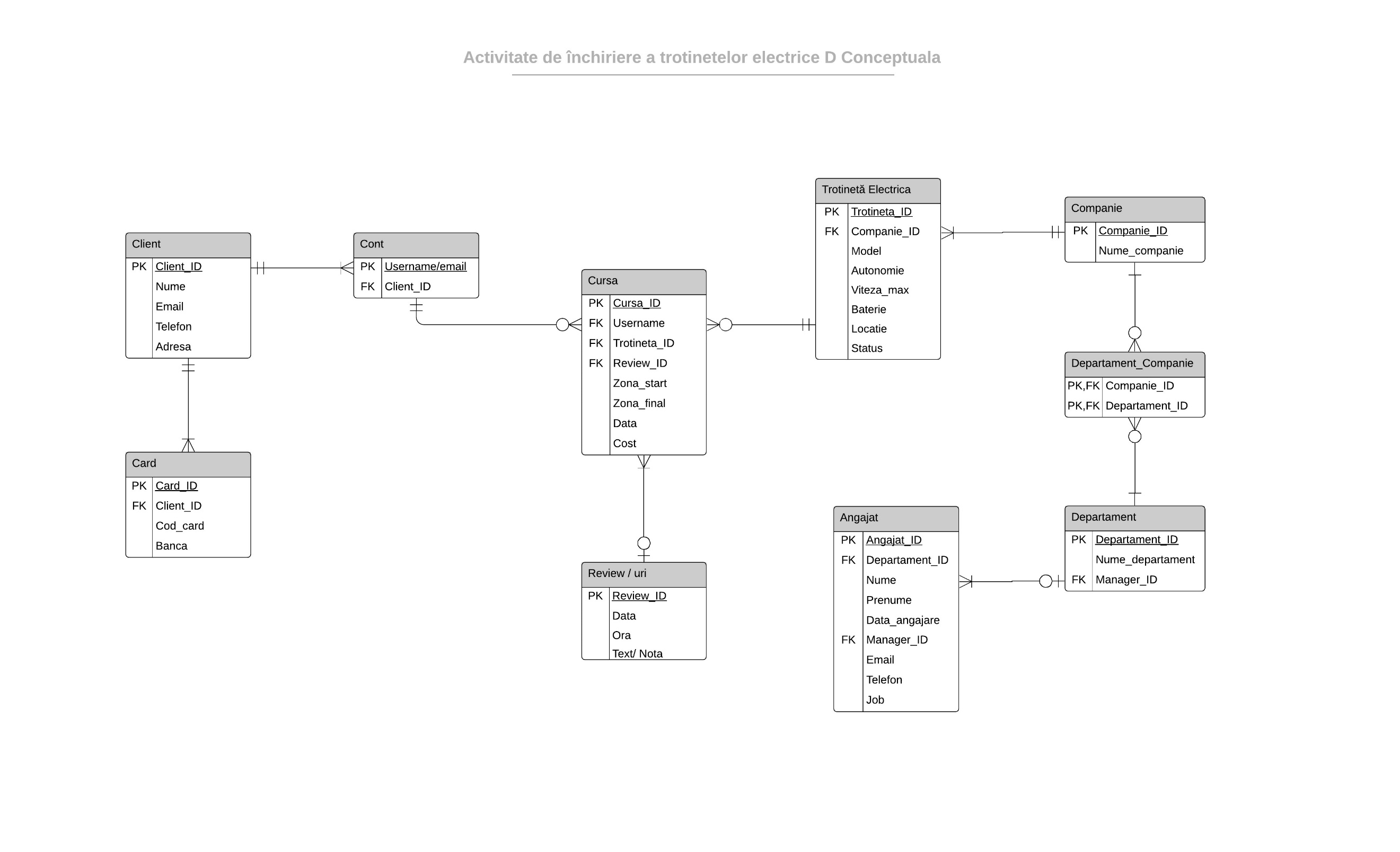
-Companie\_id not null,foreign key + on delete cascade

-Locatie not null

-Baterie check > 0 & < 100

-Status default „Disponibila”

Trotineta\_id primește primary key pentru că este atributul principal din tabel și care trebuie să fie pozitiv (check > 0). Atributul Companie\_id primește foreign key doarece face legătura cu tabelul Companie. Următoarele constrângeri sunt not null pentru Locație, doarece trotineta trebuie gasită într-un loc, check > 0 & < 100 pentru baterie, reprezentând un procentaj, și default ca „Disponibila” pentru status-ul trotinetei.



Tabelul **Cursa** conține la nivel de constrângeri:

-Cursa\_id primary key

-Username not null, foreign key + on delete cascade

-Trotineta\_id foreign key + on set null

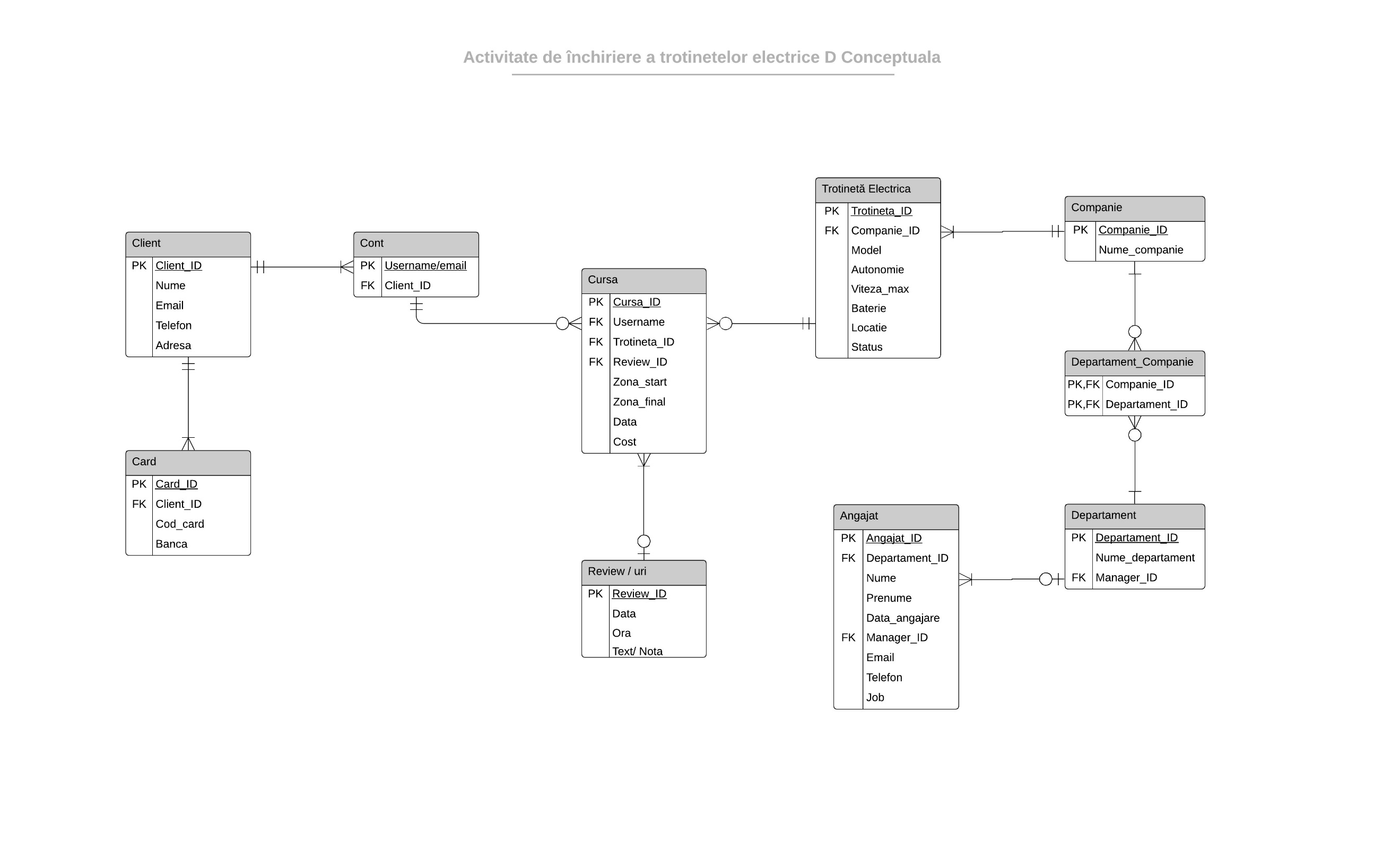
-Review\_id foreign key + on set null

-Zona start not null

-Zona final not null

-Cost default 3

Primary key-ul tabelului este oferit id-ului cursei, fiind atributul principal, iar foreign key-urile aparțin Username-ului (legatura cu Cont) , Trotineta\_id-ului (legatura cu Trotineta\_electrica) și Review\_id-ului (legatura cu tabelul Review). Atributele Zona\_start și Zona\_final primesc constrângerea de not null doarece clientul trebuie să menționeze din ce locație pană în ce locație închiriază trotineta. Cost primește default 3 doarece atât costă deblocarea unei trotinete chiar dacă se parcurge o distanță sau nu.

Tabelul **Angajat** deține urmatoarele constrângeri:

-Angajat\_id primary key, check > 0

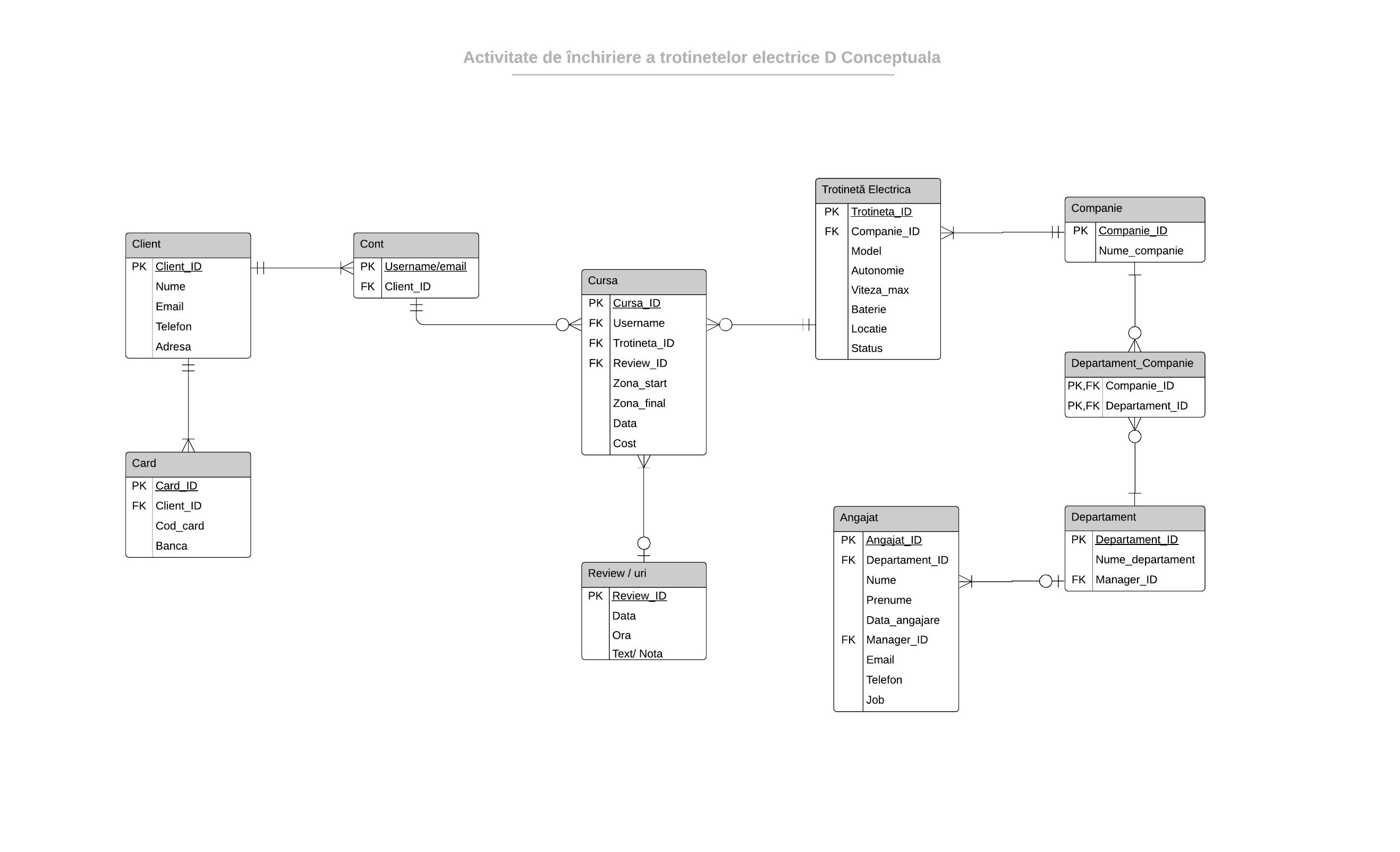
-Departament\_id foreign key + on delete cascade, not null, check > 0

-Nume not null

-Prenume not null

-Manager\_id foreign key

Atributul Angajat\_id primește primary key și check > 0 deoarece e coloana principală în tabel. Foreign key-ul este dat Departament\_id-ului doarece acesta reprezintă legătura cu tabelul Departament și care trebuie în fișa angajatului să fie diferit de null. Atributele Nume, Prenume primesc constrângerea de tip not null întrucât un angajat trebuie să aibă nume și prenume, iar manager\_id are foreign key din motiv că un angajat poate să aibă un manager, care e la randul lui un angajat, făcând legătura cu același tabel.

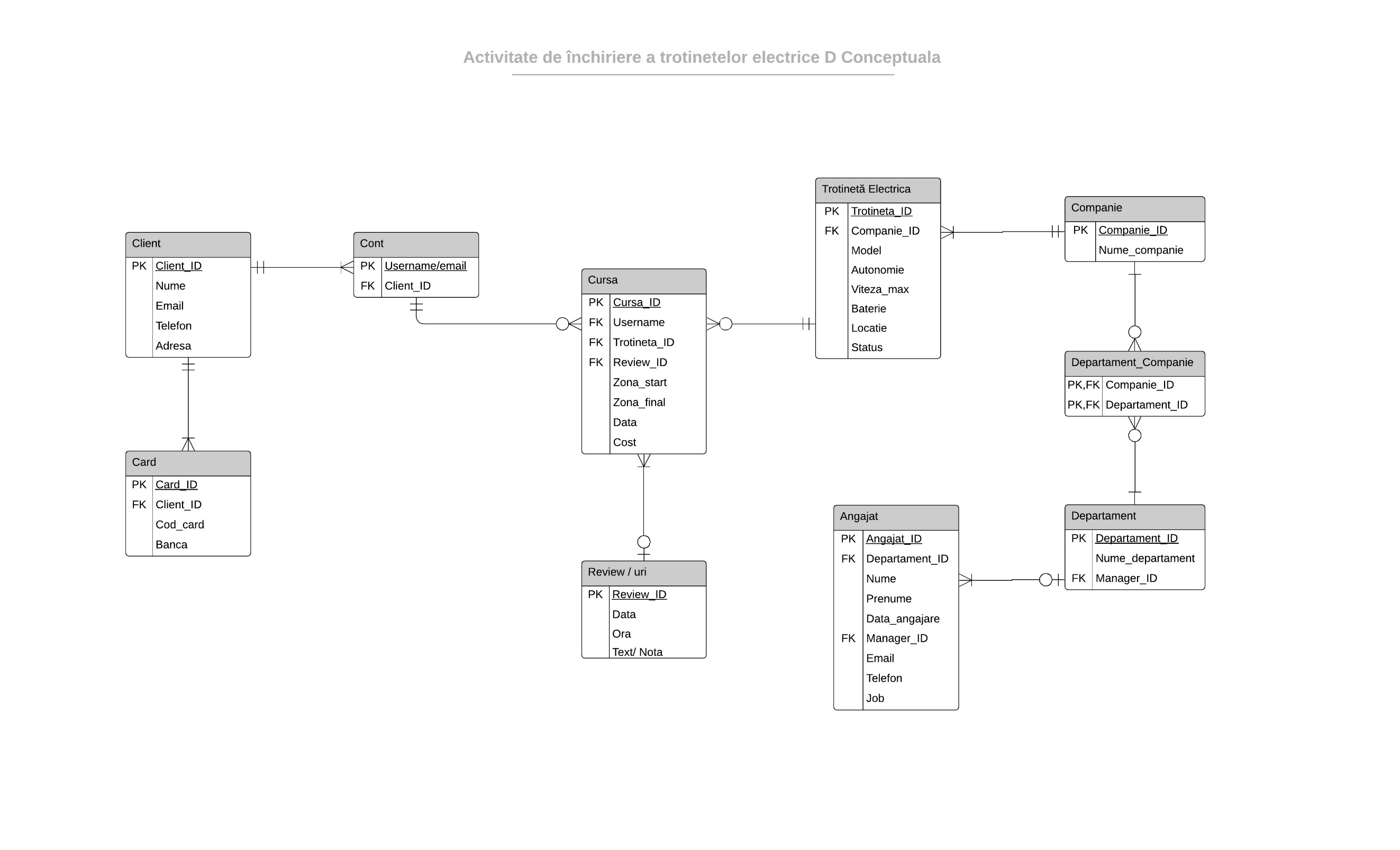
Tabelul **Departament** are ca și constrângeri:

-Departament\_id primary key, check > 0

-Nume\_departament not null, unique

-Manager\_id foreign key

Tabelul conține constrângere de tip primary key pe coloana Departament\_id, fiind coloana principală și care are în plus check > 0 pentru un id pozitiv. Foreign key-ul este pe atributul Manager\_id, acesta reprezentând legatura cu tabelul Angajat. Numele departamentului primește constrângeri de tip not null și unique doarece un departament trebuie să aibă neapărat un nume unic.

 Tabelul **Departament\_Companie** conține următoarele constrângeri:

-Companie\_id foreign key

-Departament\_id foreign key

Tabelul fiind unul asociativ ca urmare a unei legaturi many-to-many are primary key-ul și foreign key-ul atât pe atributul Companie\_id, cât și pe Departament\_id.

6. SCHEMELE RELAȚIONALE (DESCRIEREA CONSTRÂNGERILOR ON DELETE CASCADE / SET NULL)

on cascade

on cascade

Client Cont Trotineta\_el Companie

set null

on cascade

on cascade

Cursa Departament\_companie

set null

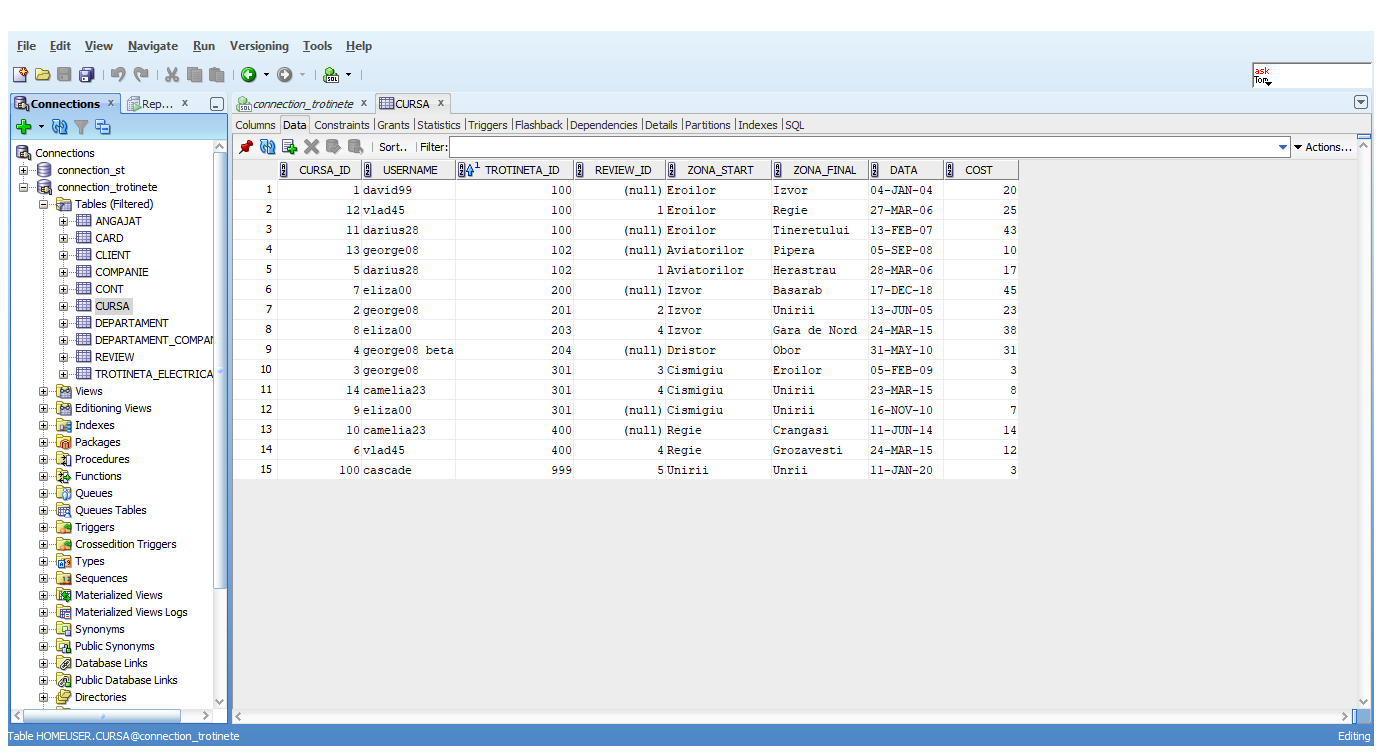
Card

Review Departament

on cascade

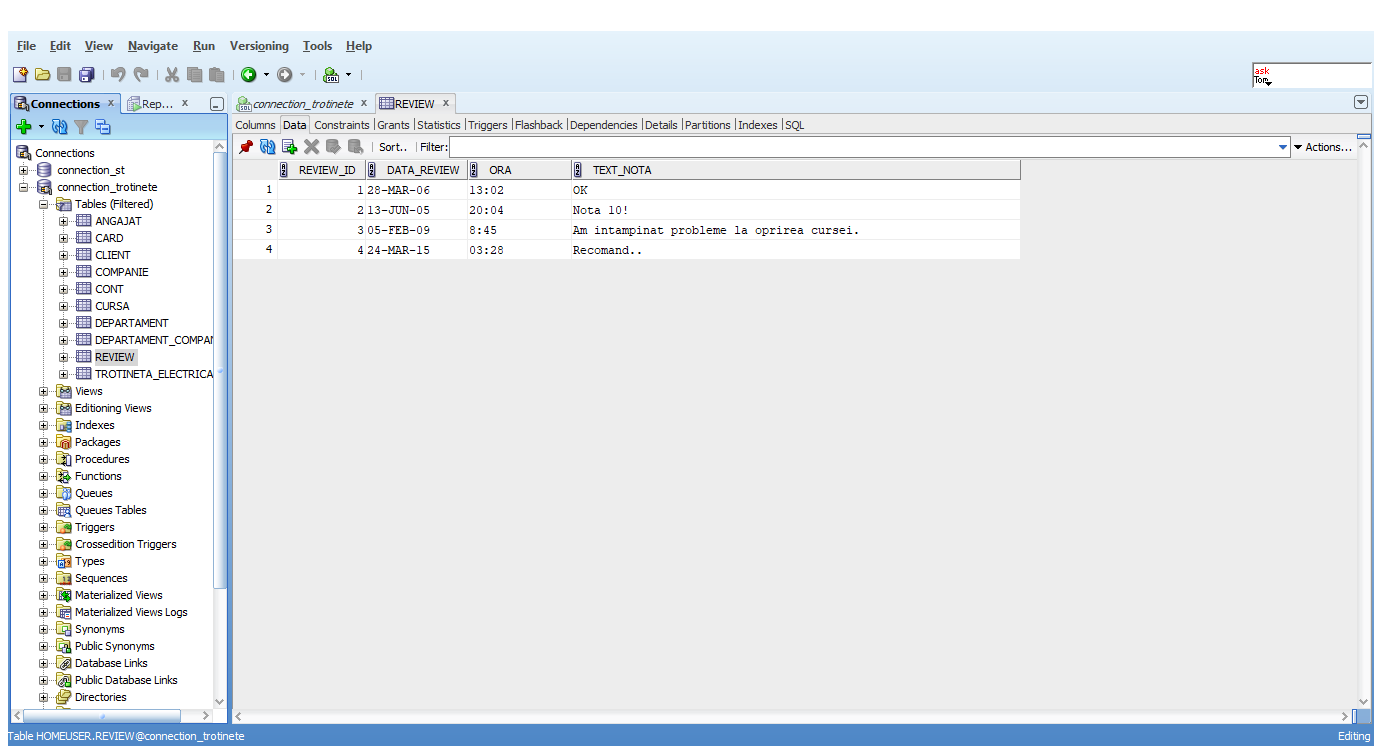
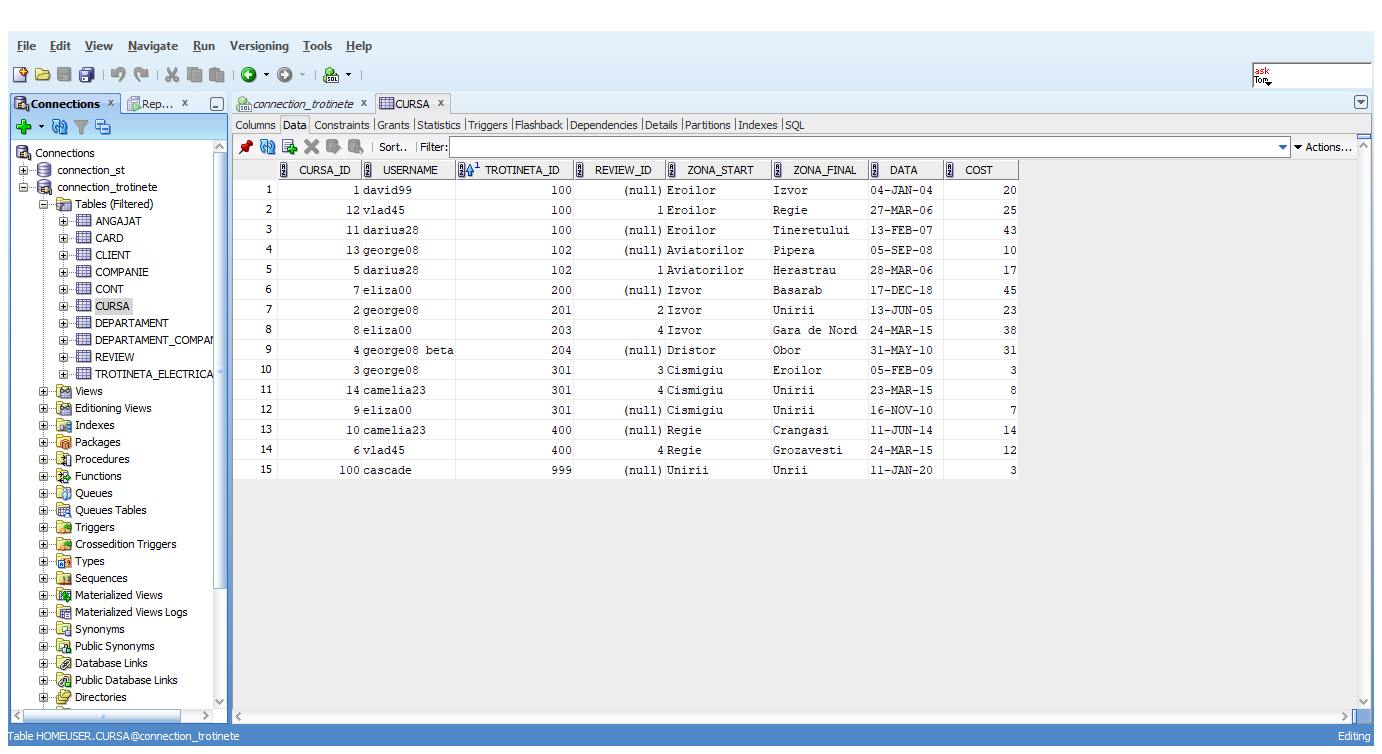
Angajat

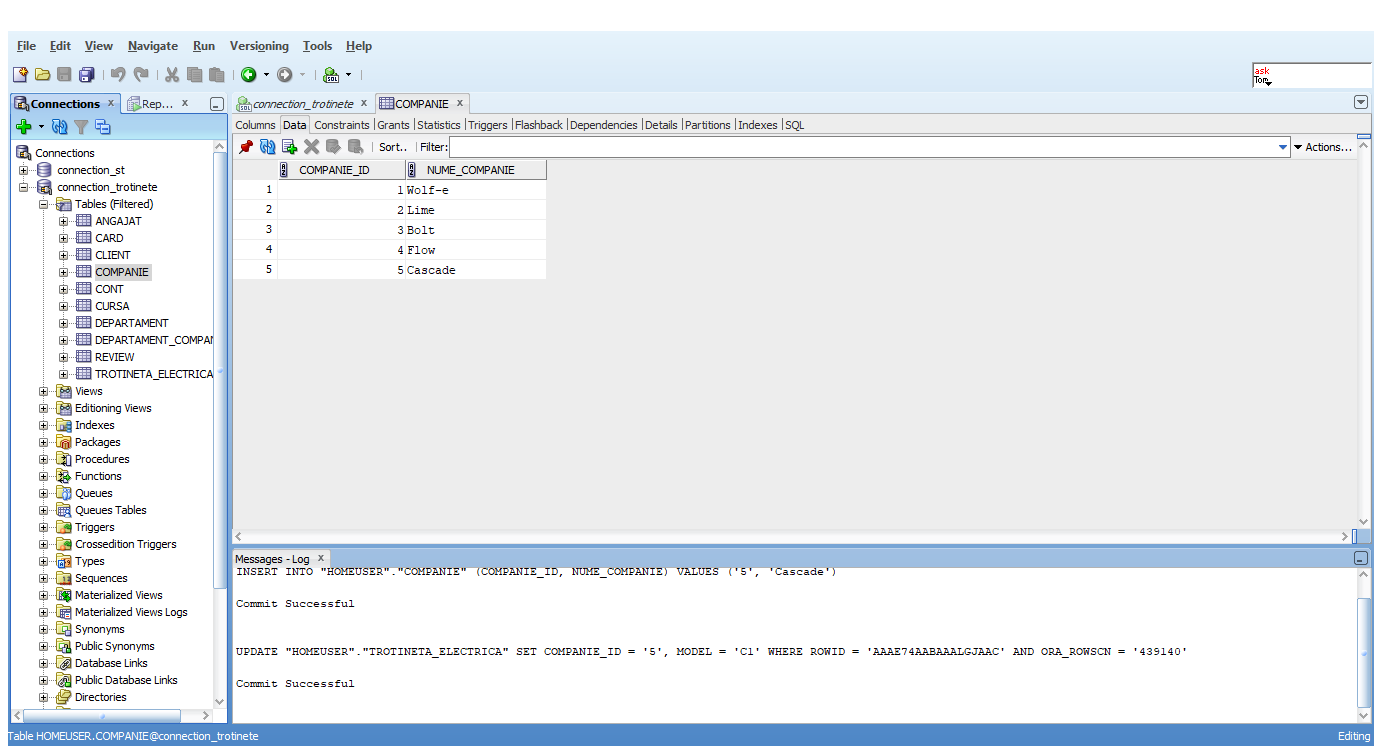
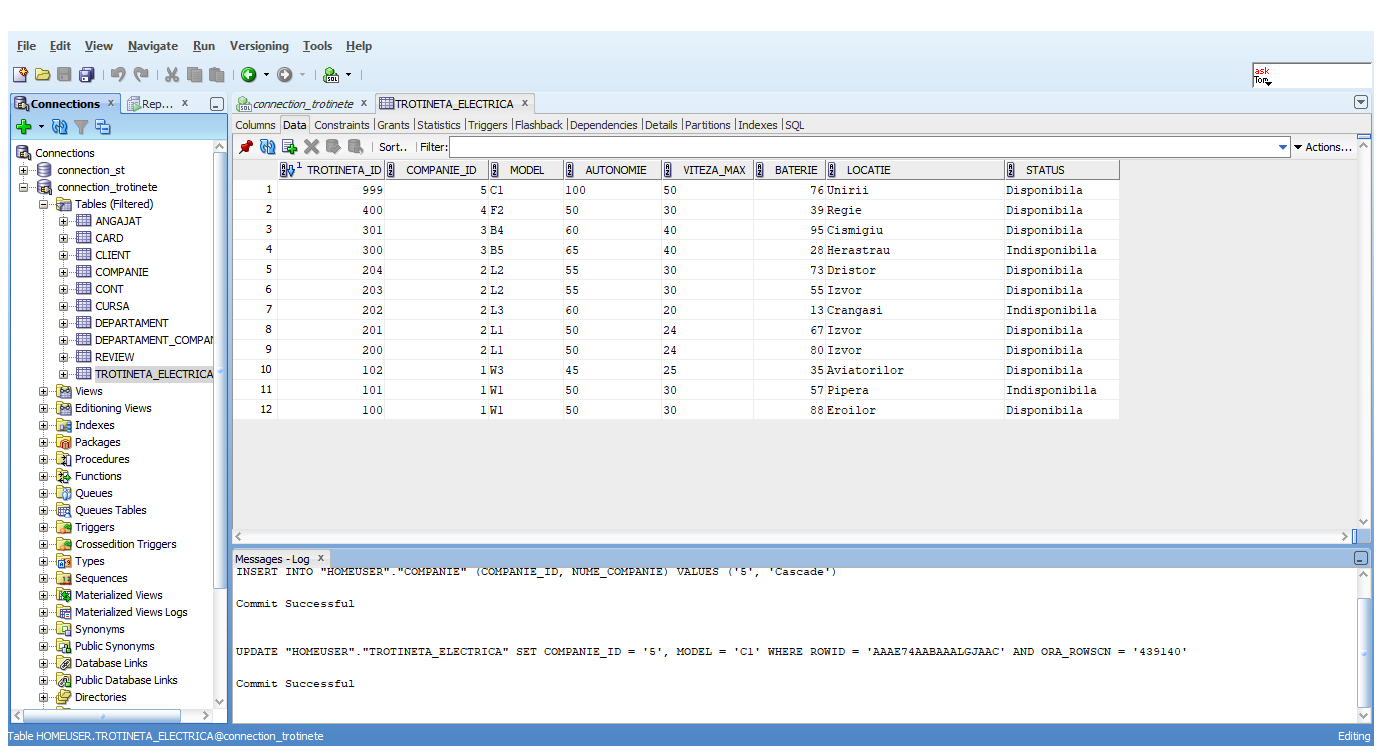
* Prin ștergerea unui review în tabelul **Cursa** în loc de id-ul review-ului apare null.



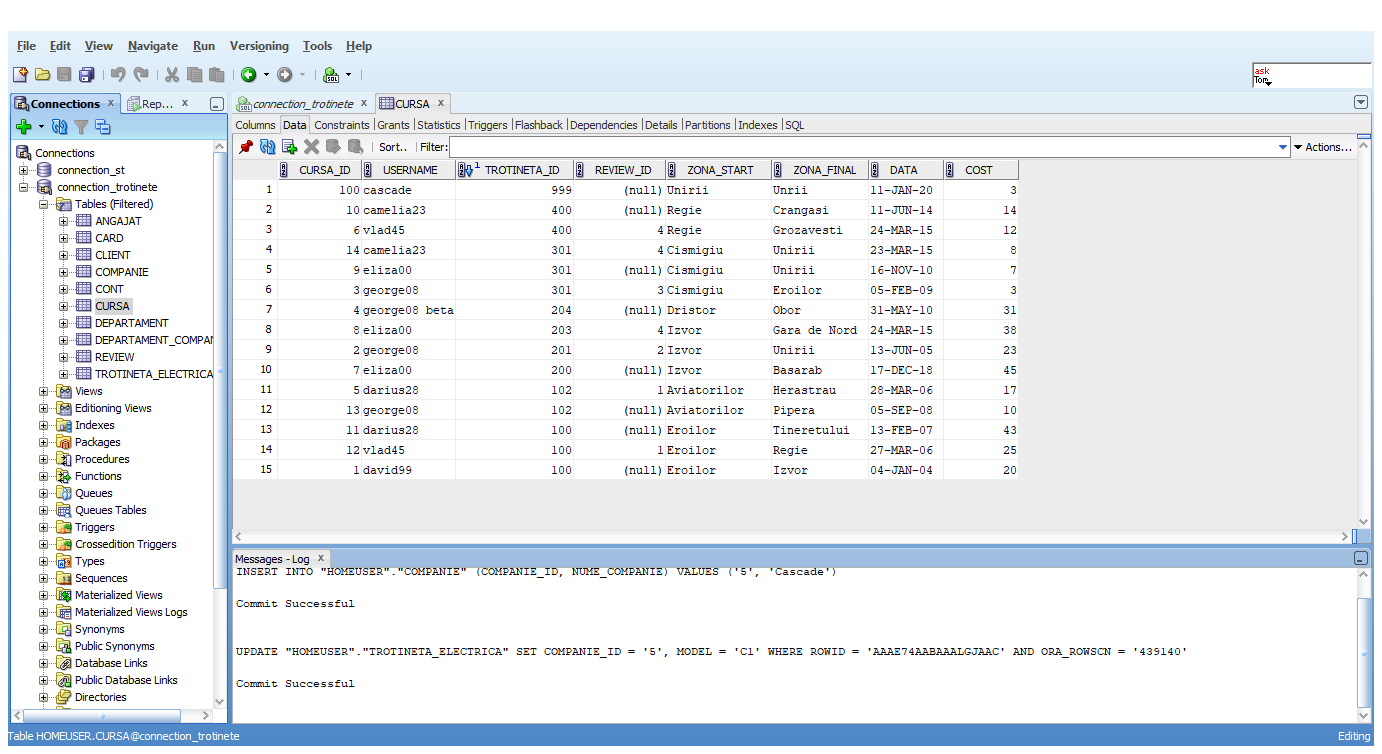
delete from review

where review\_id = 5;

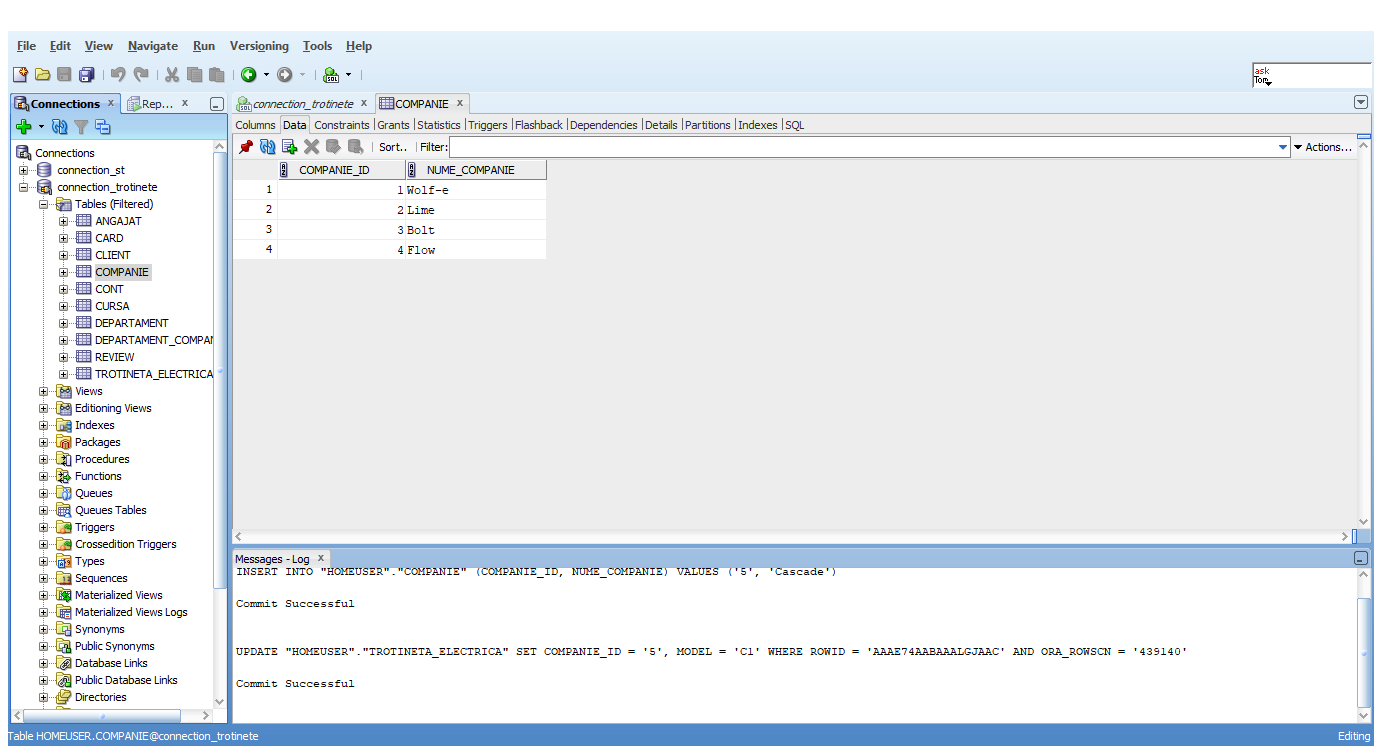
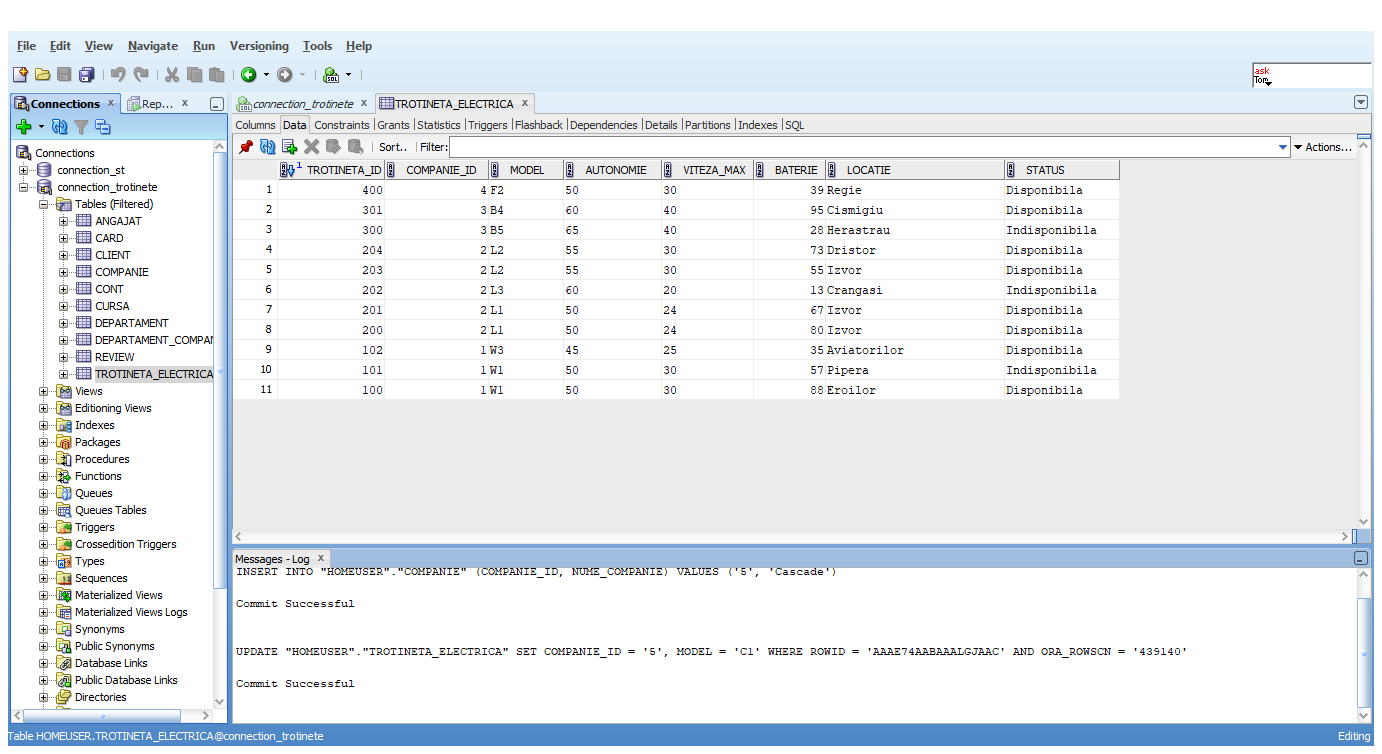


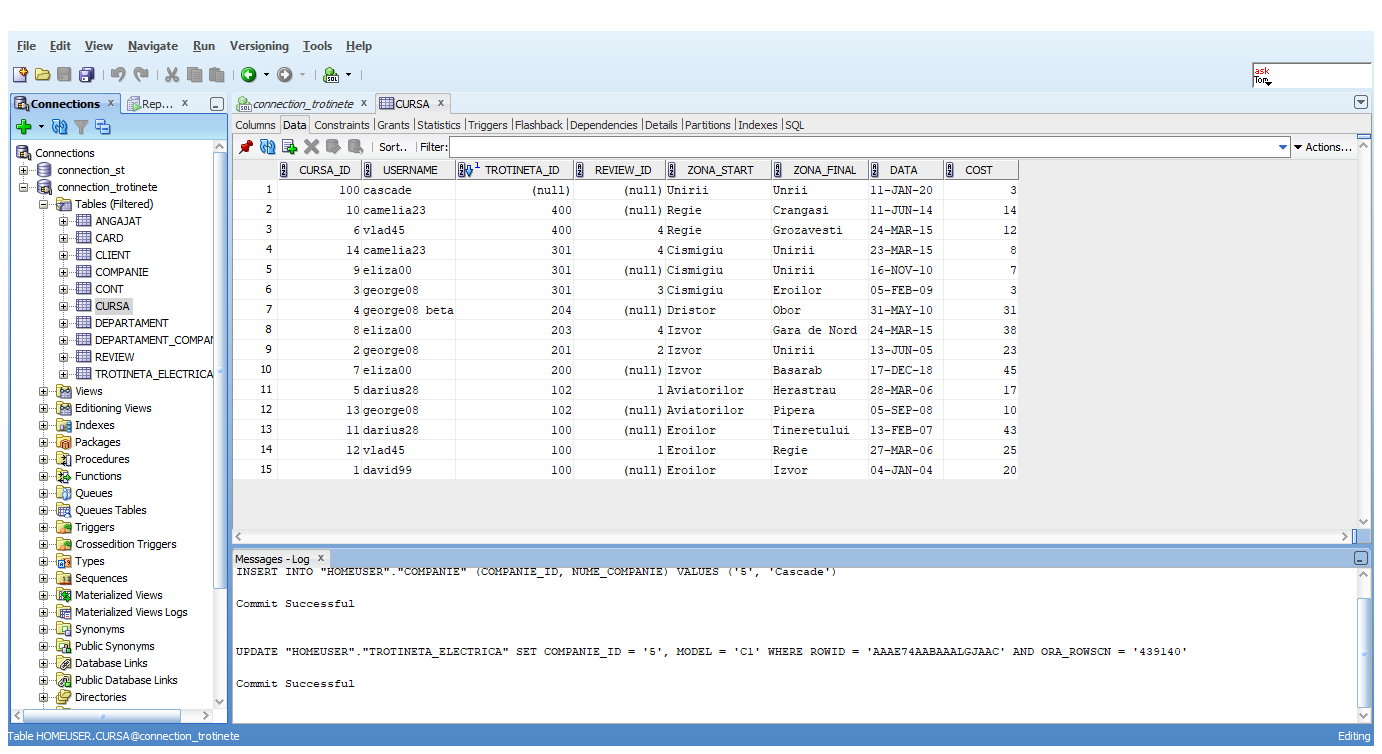
* Prin ștergerea unei companii se șterg trotinetele ce apartin de ea, iar din cauză

că se scot trotinetele, în tabelul **Cursa** în locul trotinetelor se trece null.

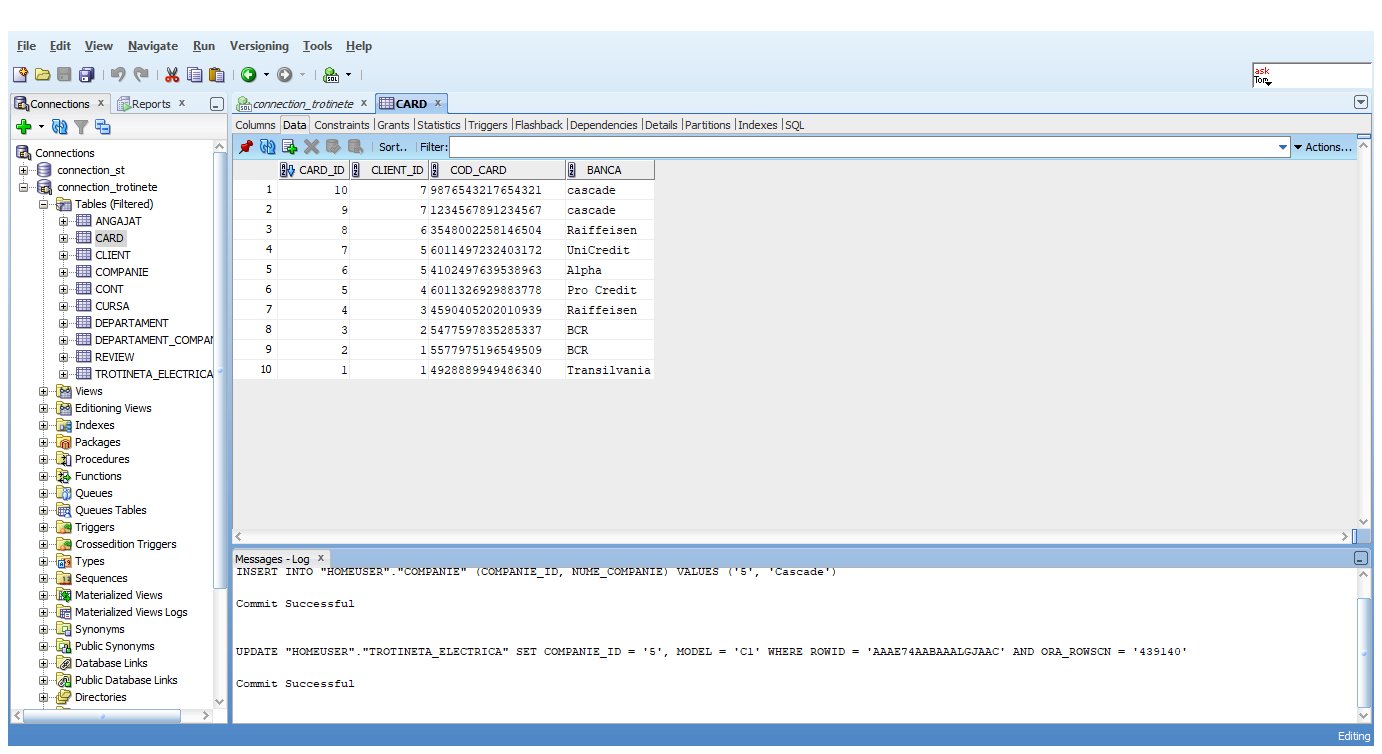
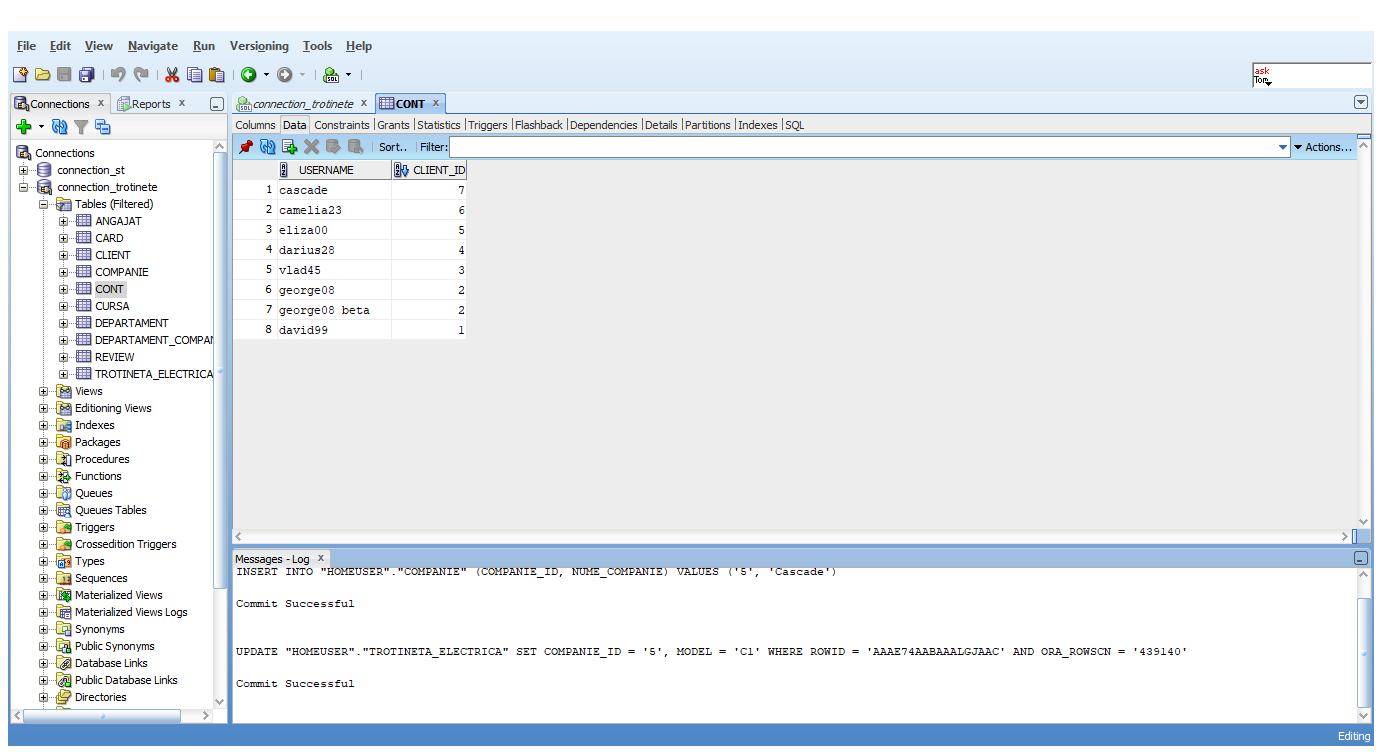


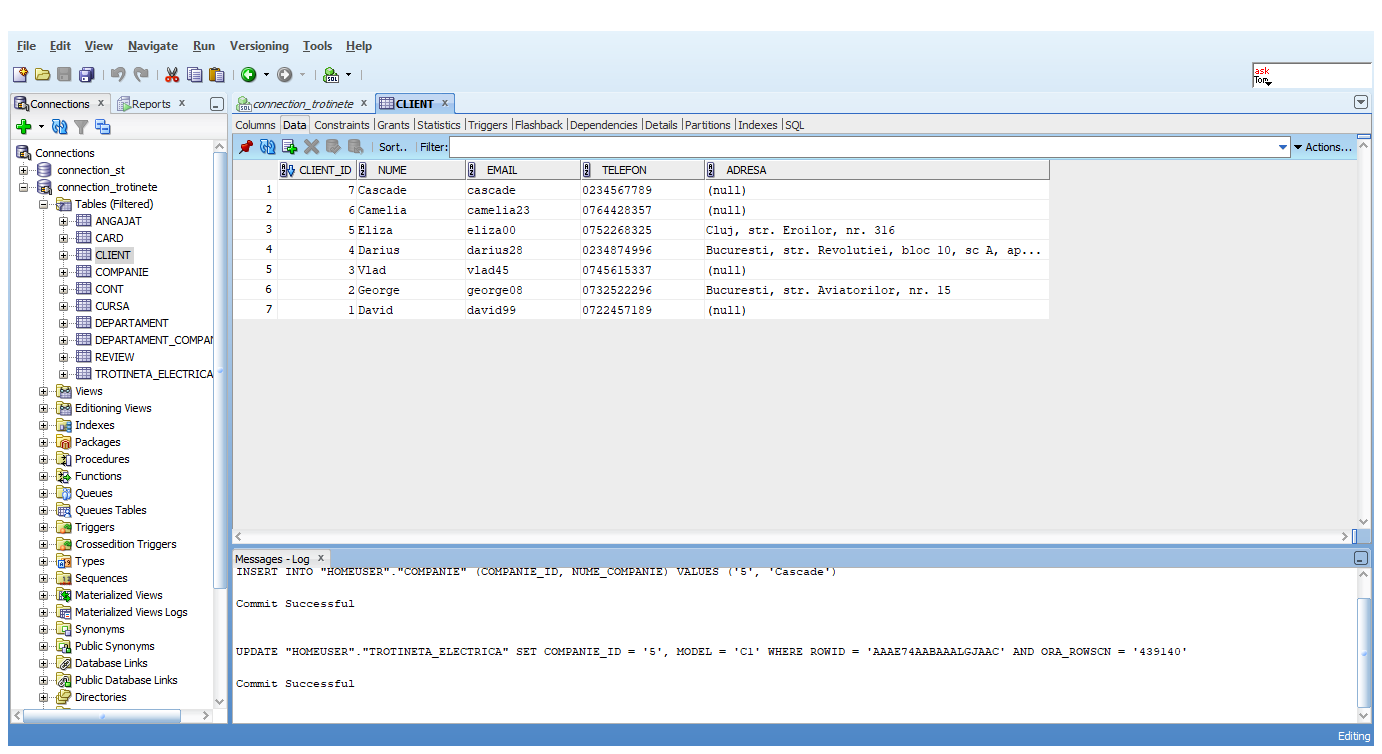
delete from companie where companie\_id = 5;

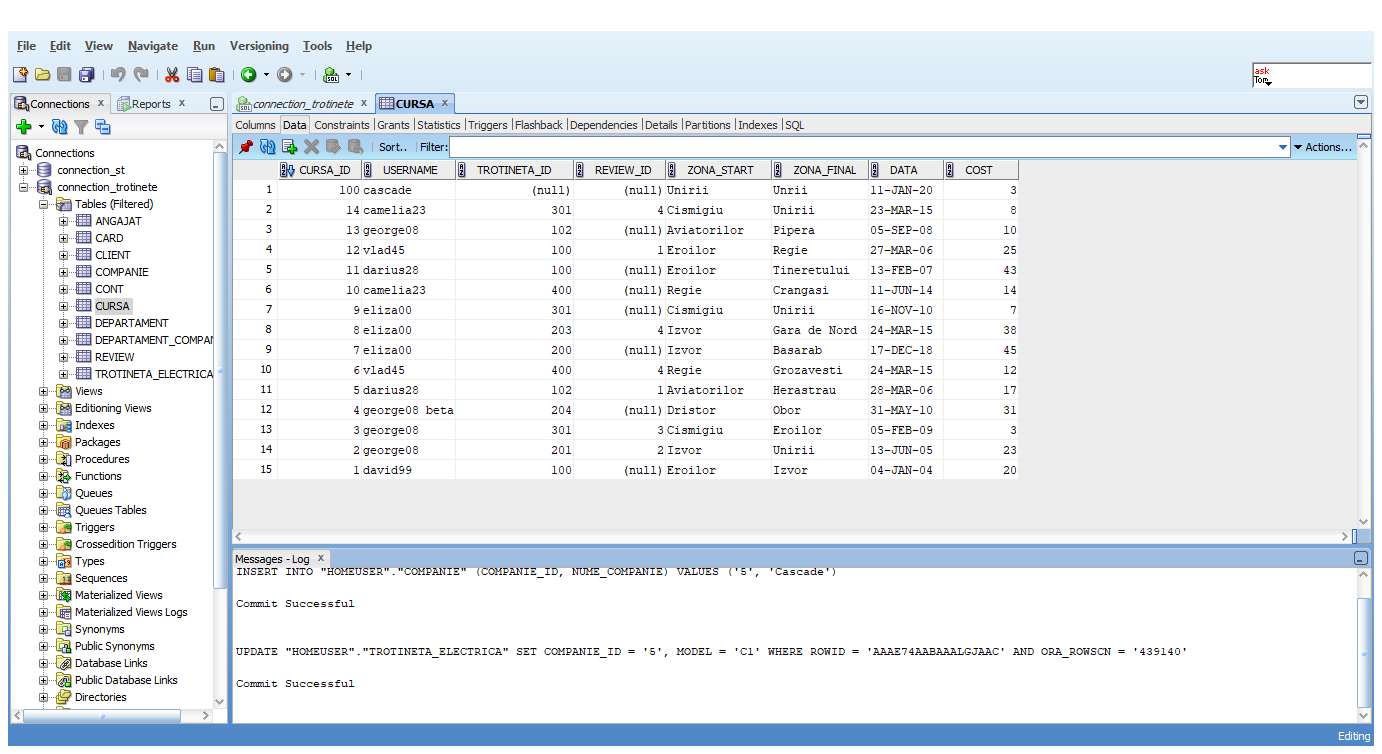




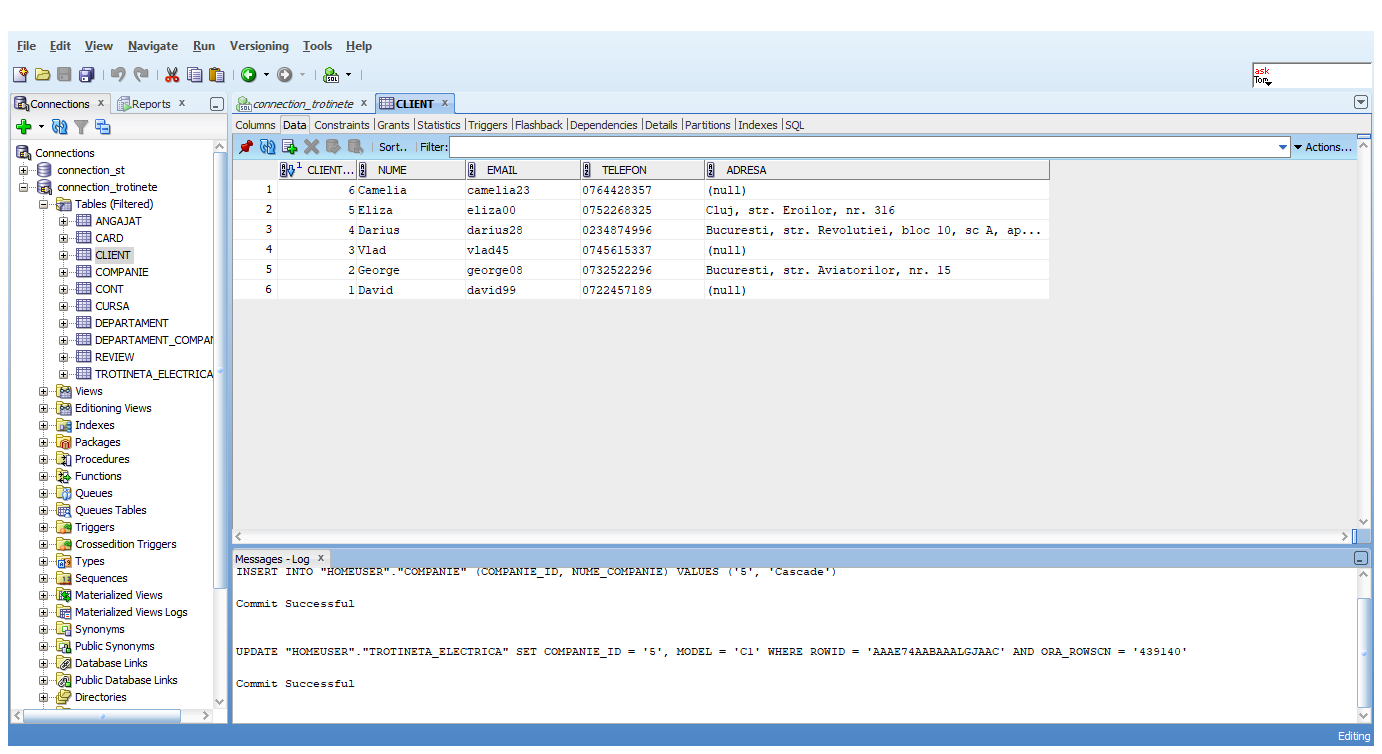
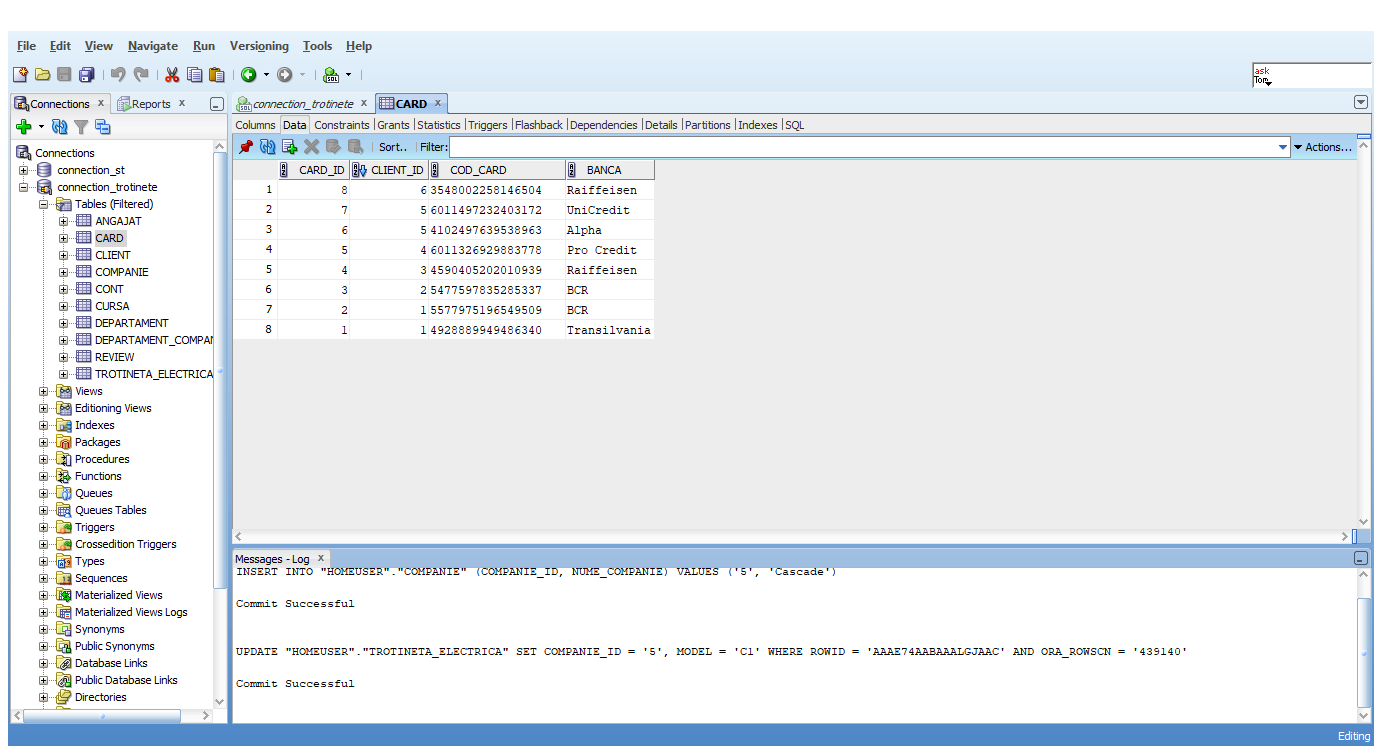
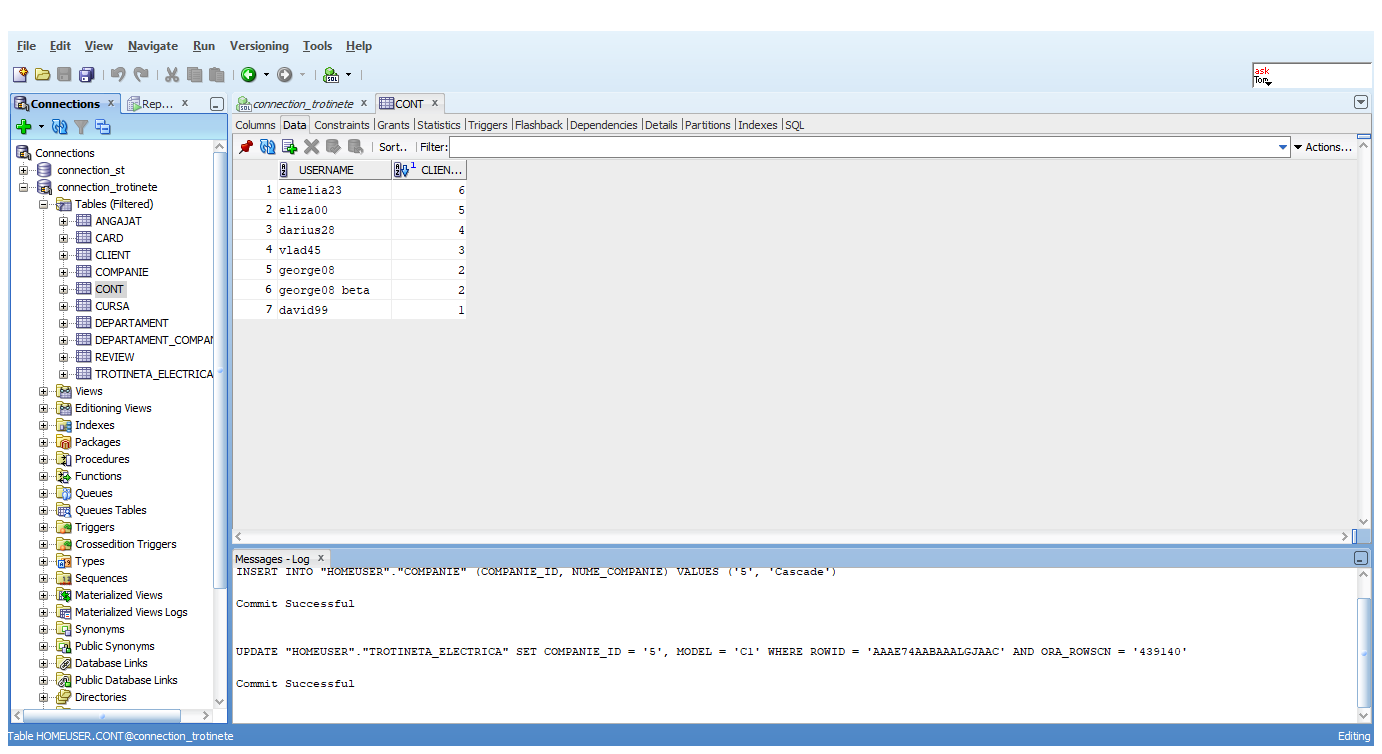
* Datorită constrângerii de tip on cascade la ștergerea unui client se șterg atât cardurile

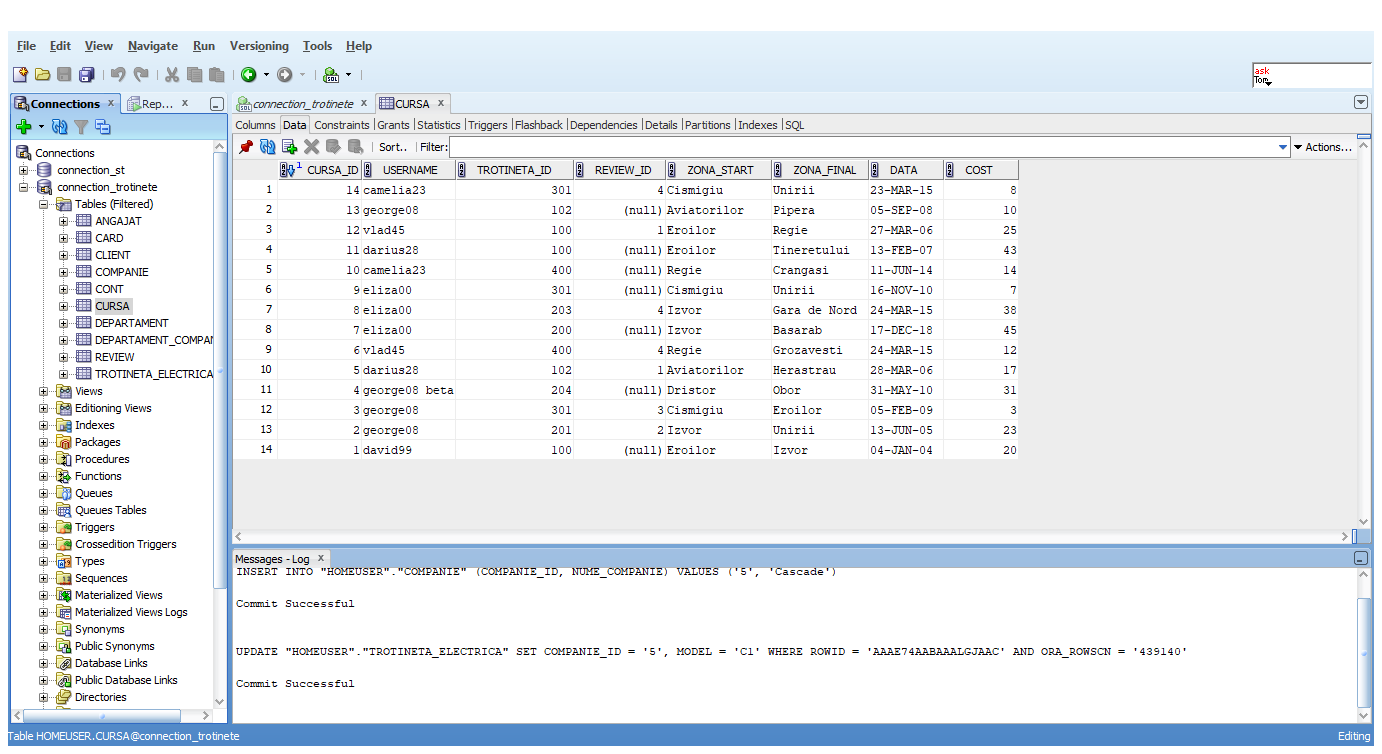
acestuia cât și conturile lui. Prin ștergerea contului se șterg și cursele care aparțin de acesta.





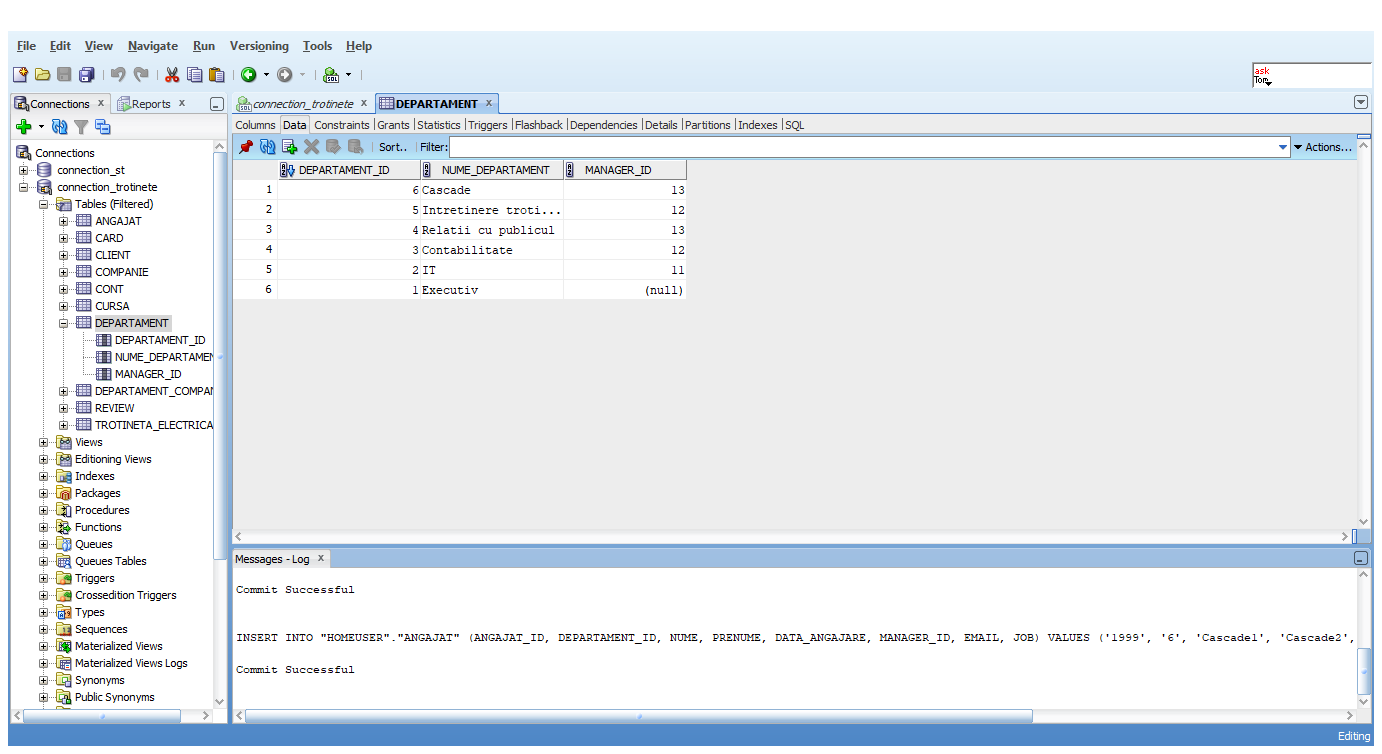
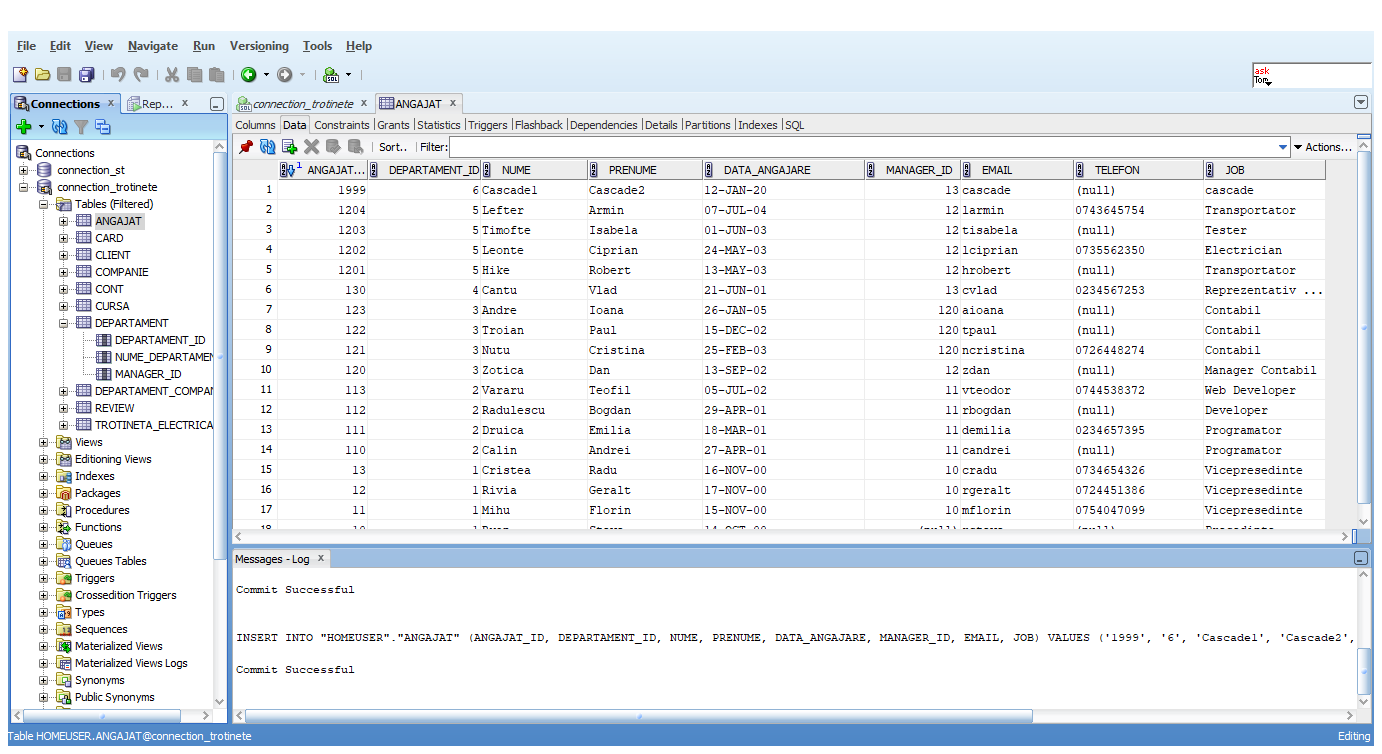
delete from client where client\_id = 7;



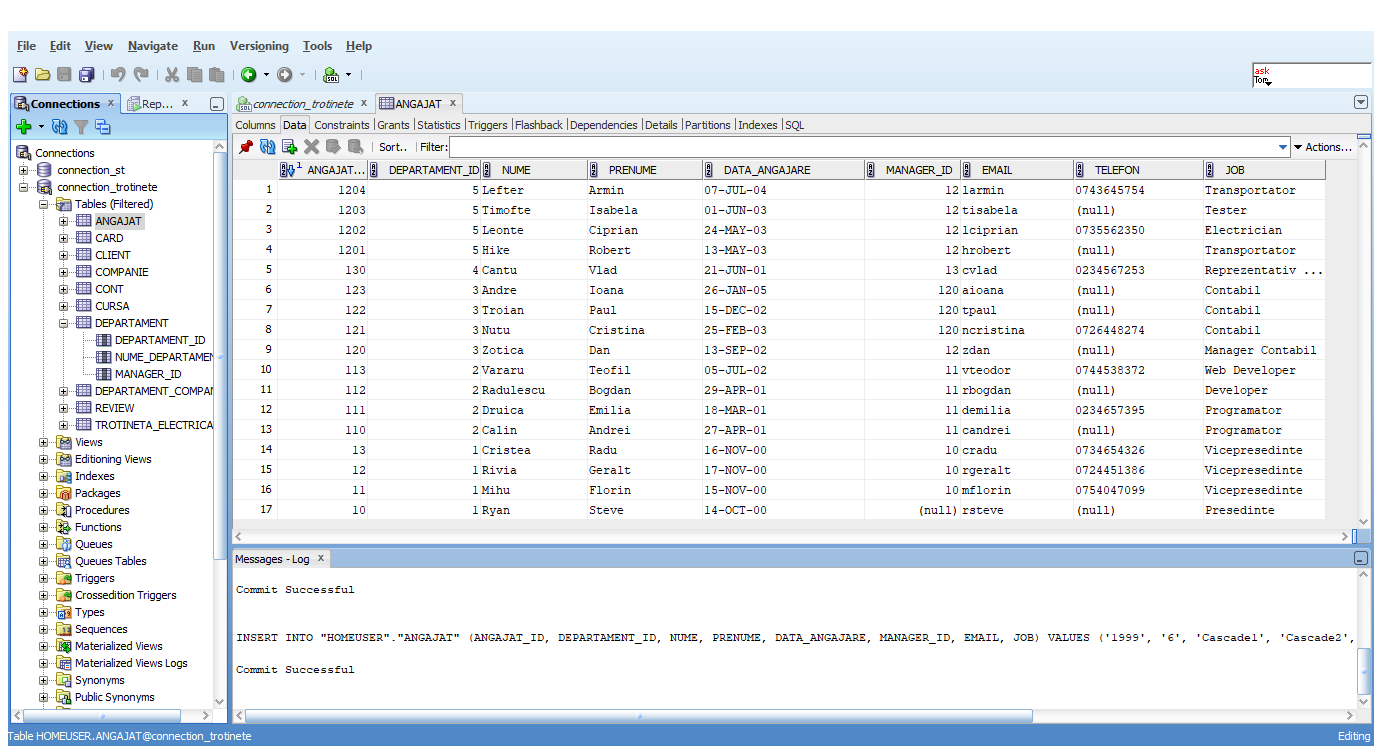


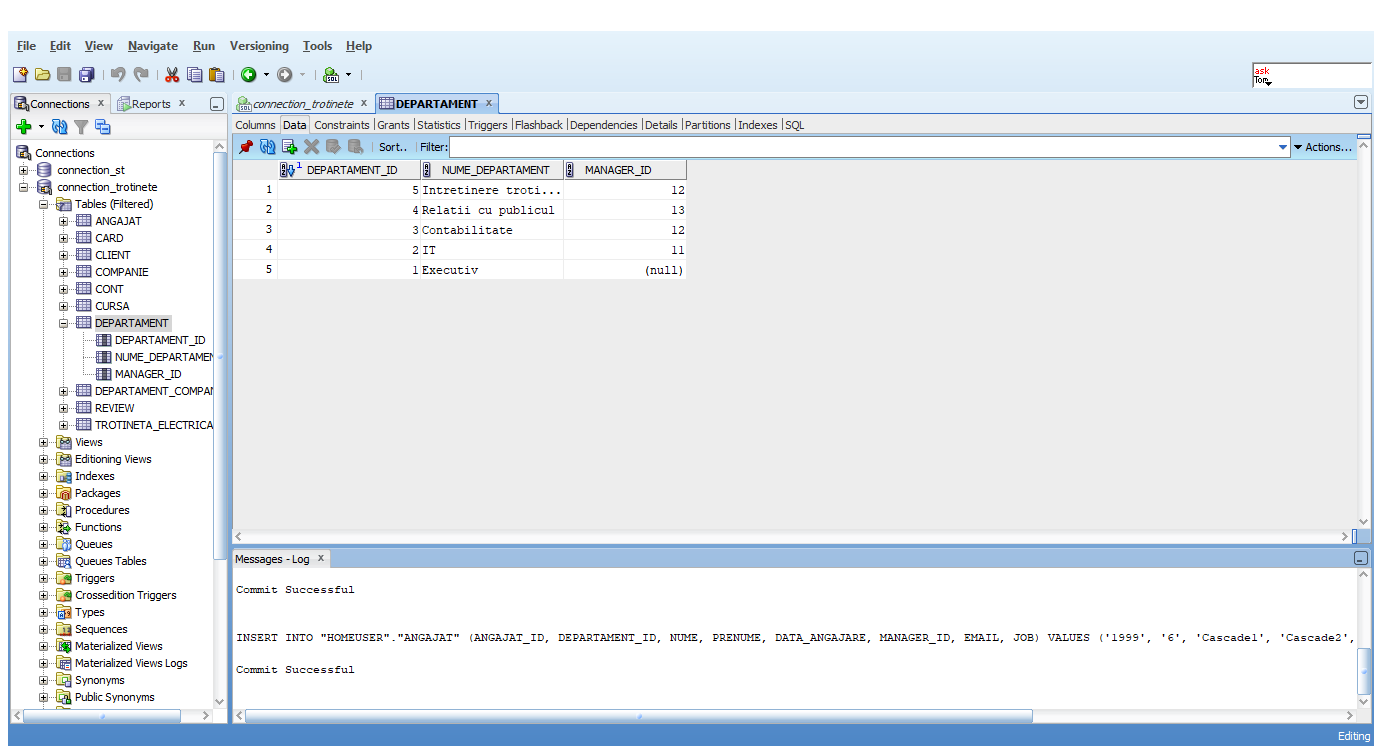
* Având la departament constrângerea de on delete cascade atunci când ștergem un

departament se șterg automat și angajații din acel departament.



delete from departament where departament\_id = 6;





Observații: Pentru ștergerea unui departament sau a unei companii si a ilustra constrângerile on delete cascade sau set null acestea nu trebuie să apară în tabelul Departament\_Companie (deoarece ambele sunt chei primare în tabel, astfel că stergerea lor duce la eroare).

7. CREAREA TABELELOR (INCLUSIV A CONSTRÂNGERILOR)

În cadrul proiectului serverul local este realizat cu Oracle Database 11g, iar conectarea la serverul bazei de date se face cu Oracle SQL Developer.

Tabel **Client**:

Create table Client (

Client\_id number(10) constraint client\_pk primary key,

Nume varchar2(50) constraint verif\_nume not null,

Email varchar2(50) constraint verif\_email not null,

Telefon varchar2(50) constraint verif\_tel not null,

Adresa varchar2(50),

constraint client\_id\_poz check (client\_id > 0)

);

Tabel **Card**:

create table Card (

Card\_id number(10) constraint card\_pk primary key,

Client\_id number(10) not null,

Cod\_Card varchar2(20) constraint verif\_cod\_card not null,

Banca varchar2(20),

constraint client\_fk foreign key (Client\_id)

references Client(Client\_id) on delete cascade,

constraint card\_id\_poz check (card\_id > 0)

);

Tabel **Cont**:

create table Cont (

username varchar2(20) constraint usr\_pk primary key,

client\_id number(10) not null,

constraint client\_fk2 foreign key (client\_id)

references Client(client\_id) on delete cascade

);

Tabel **Review**:

create table review (

Review\_id number(10) constraint review\_pk primary key,

Data\_review date default sysdate,

Ora varchar2(10) constraint verif\_ora not null,

Text\_Nota varchar2(50),

constraint review\_id\_poz check (review\_id > 0)

);

Tabel **Companie**:

create table companie (

companie\_id number(10) constraint companie\_pk primary key,

nume\_companie varchar2(20) constraint verif\_nume\_comp not null,

constraint companie\_id\_poz check (companie\_id > 0),

constraint nume\_companie\_uk unique (nume\_companie)

);

Tabel **Trotineta\_electrica**:

create table trotineta\_electrica (

trotineta\_id number(10) constraint trotineta\_pk primary key,

companie\_id number(10) not null,

model varchar2(20),

autonomie varchar2(10),

viteza\_max varchar2(10),

baterie number(10),

locatie varchar2(20) constraint verif\_loc not null,

status varchar2(20) default 'Disponibila',

constraint companie\_fk foreign key (companie\_id)

references companie(companie\_id) on delete cascade,

constraint trotineta\_id\_poz check (trotineta\_id > 0),

constraint trotineta\_baterie\_poz check (baterie > 0),

constraint bat\_full check (baterie <= 100)

);

Tabel **Cursa**:

create table Cursa (

cursa\_id number(10) constraint cursa\_pk primary key,

username varchar2(20) not null,

trotineta\_id number(10),

review\_id number(10),

Zona\_start varchar2(20) constraint verif\_zst not null,

Zona\_final varchar2(20) constraint verif\_zf not null,

Data date,

Cost number(8,2) default 3,

constraint username\_fk foreign key (username)

references cont(username) on delete cascade,

constraint trotineta\_fk foreign key (trotineta\_id)

references trotineta\_electrica(trotineta\_id) on delete set null,

constraint review\_fk foreign key (review\_id)

references review(review\_id) on delete set null,

constraint cursa\_id\_poz check (cursa\_id > 0)

);

Tabel **Angajat**:

create table angajat (

angajat\_id number(10) constraint angajat\_pk primary key,

departament\_id number(10) not null,

nume varchar2(20) constraint verif\_nume\_ang not null,

prenume varchar2(20) constraint verif\_prenume\_ang not null,

data\_angajare date,

manager\_id number(10),

email varchar(50),

telefon varchar(20),

job varchar(20),

constraint angajat\_id\_poz check (angajat\_id > 0),

constraint departament\_id\_poz check (departament\_id > 0)

);

Tabel **Departament**:

create table departament (

departament\_id number(10) constraint departament\_pk primary key,

nume\_departament varchar2(50) constraint verif\_nume\_dep not null,

manager\_id number(10),

constraint manager\_fk\_dep foreign key (manager\_id)

references angajat(angajat\_id),

constraint nume\_departament\_uk unique (nume\_departament),

constraint departament\_poz check (departament\_id > 0)

);

Tabel **Departament\_companie**:

create table departament\_companie (

companie\_id number(10),

departament\_id number(10),

primary key (companie\_id, departament\_id),

constraint companie\_id\_fk2 foreign key (companie\_id)

references companie(companie\_id),

constraint departament\_id\_fk2 foreign key (departament\_id)

references departament(departament\_id)

);

Modificare facută după crearea tabelelor:

alter table angajat

add (constraint departament\_fk foreign key (departament\_id)

references departament(departament\_id) on delete cascade,

constraint manager\_fk foreign key (manager\_id)

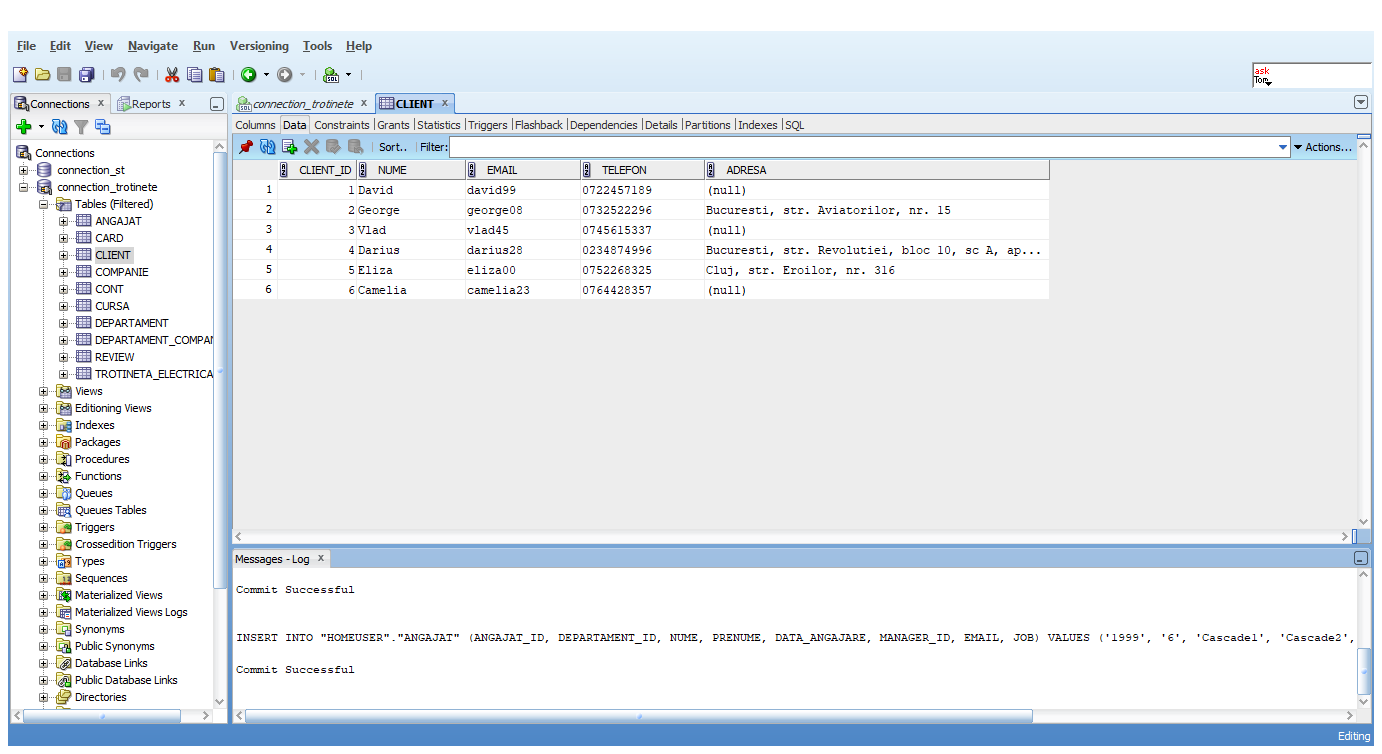
references angajat

);

8.INTRODUCERE DATE

Se prezintă cate un model de inserare pentru fiecare tabel.

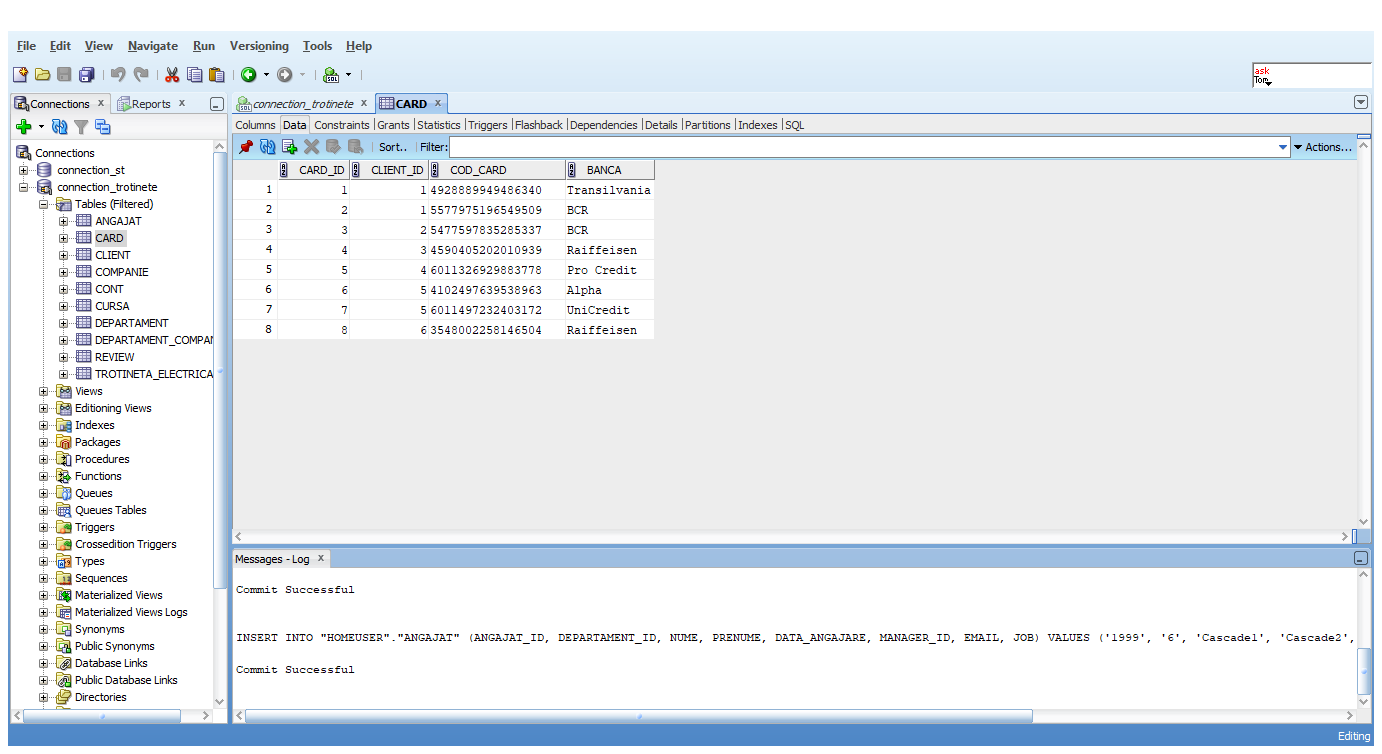
Tabel **Client**:



insert into client

values (1,'David','david99','0722457189',null);

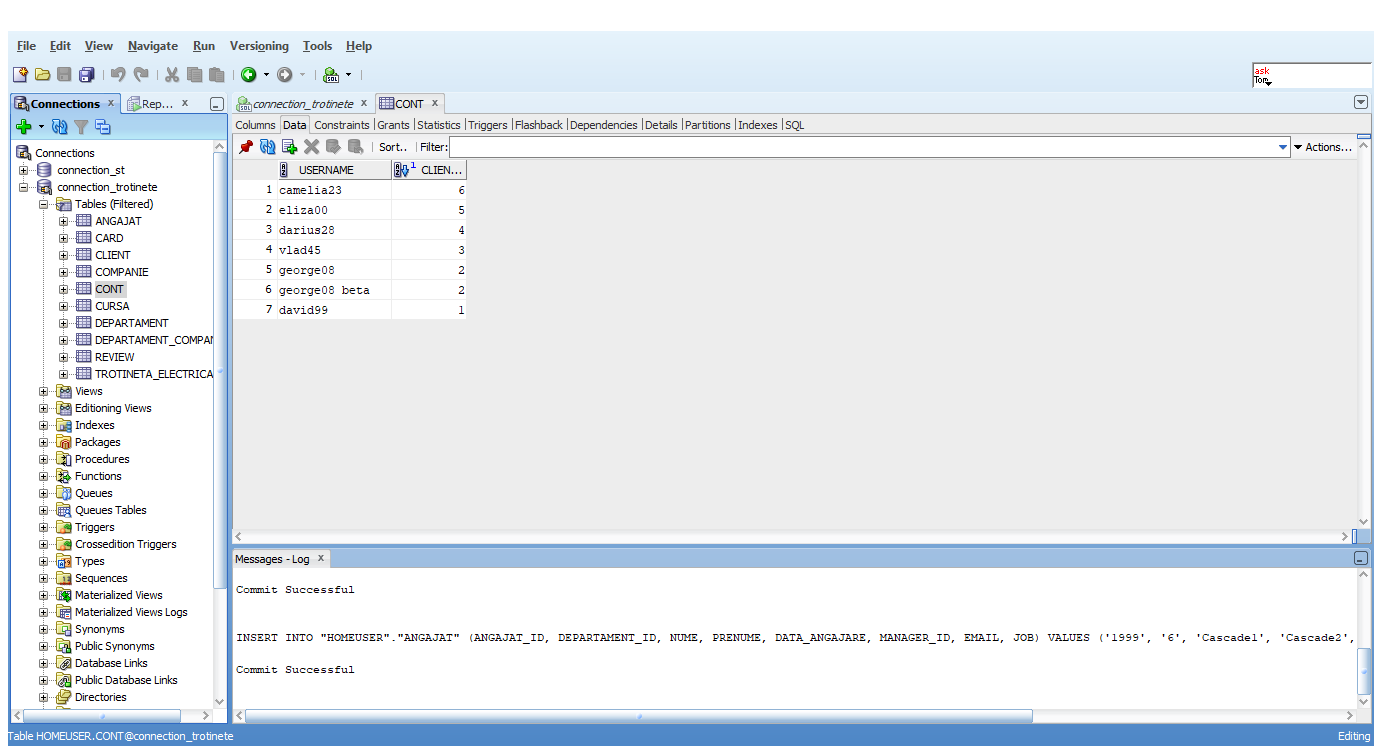
Tabel **Card**:



insert into card

values (2,1,'5577975196549509','BCR');

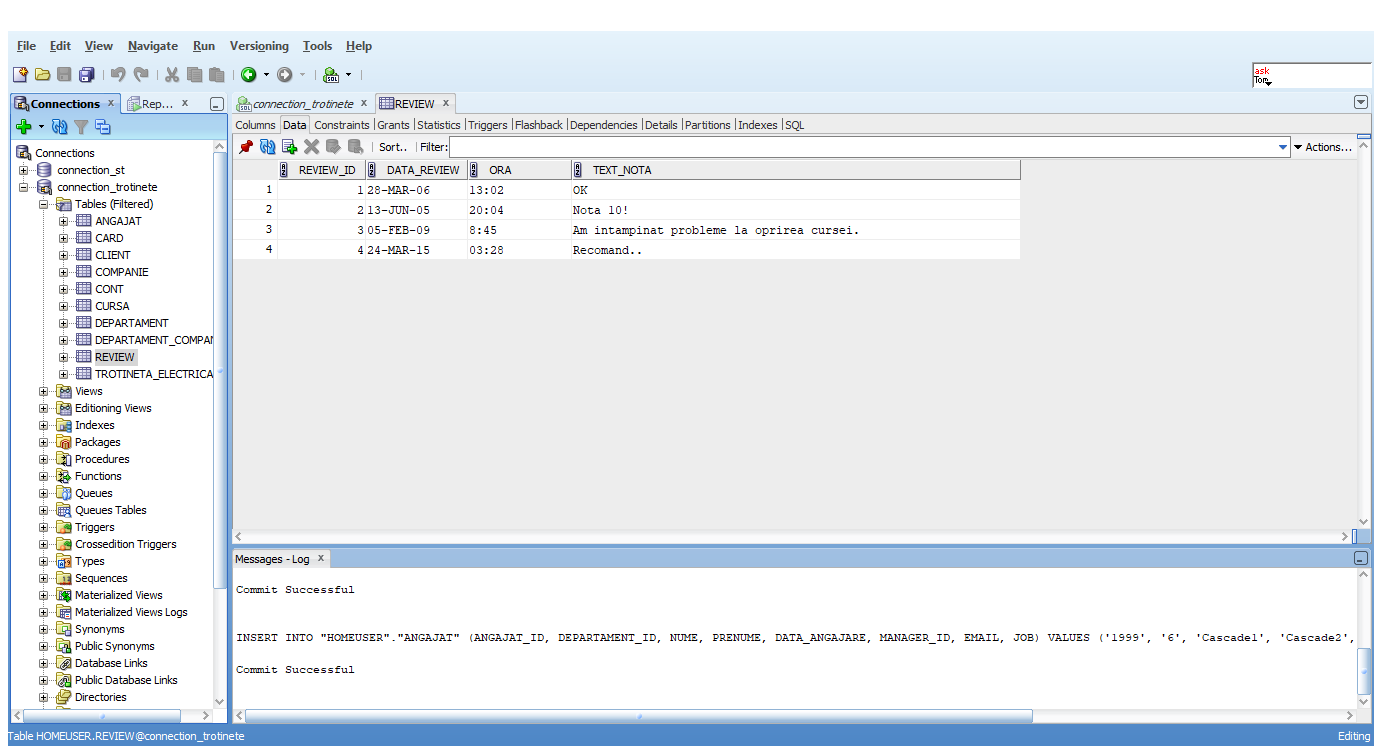
Tabel **Cont**:



insert into cont

values ('david99',1);

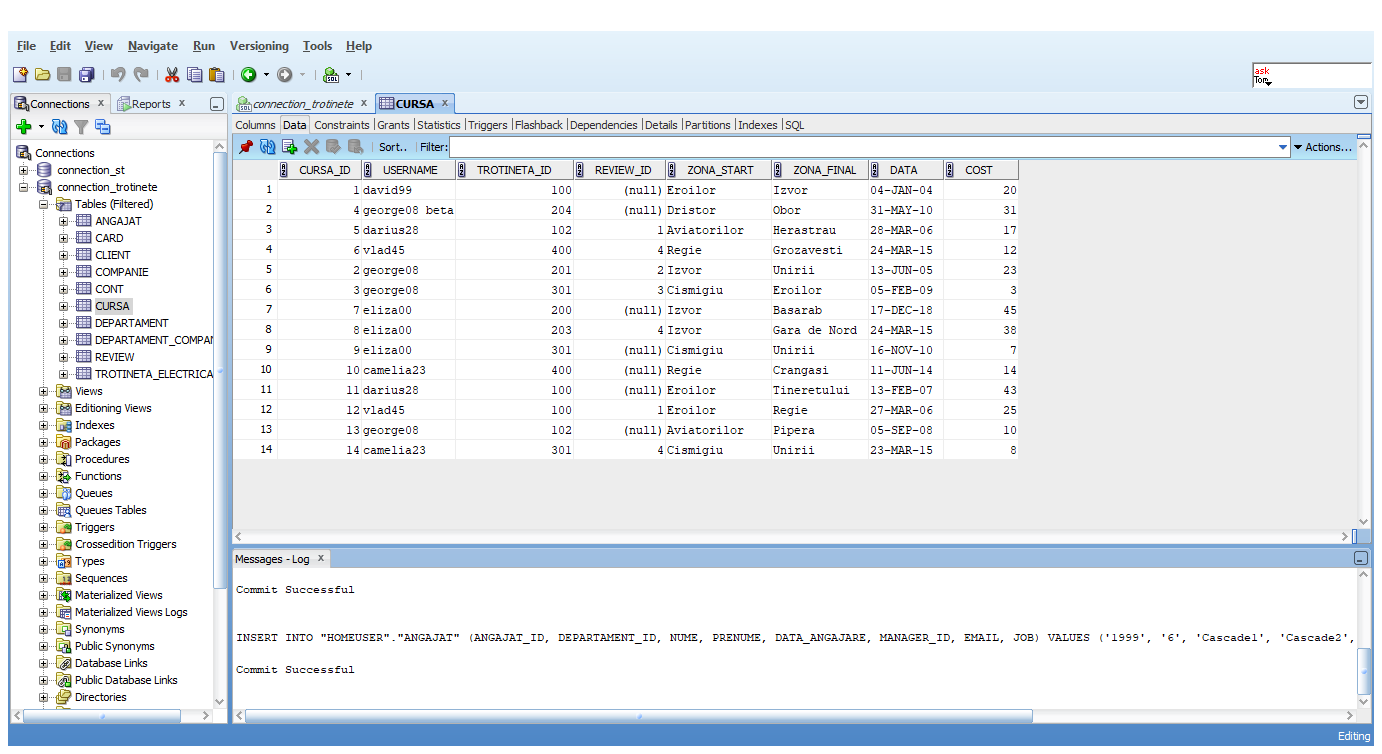
Tabel **Review**:



insert into review

values (1,'28-mar-2006','13:02','OK');

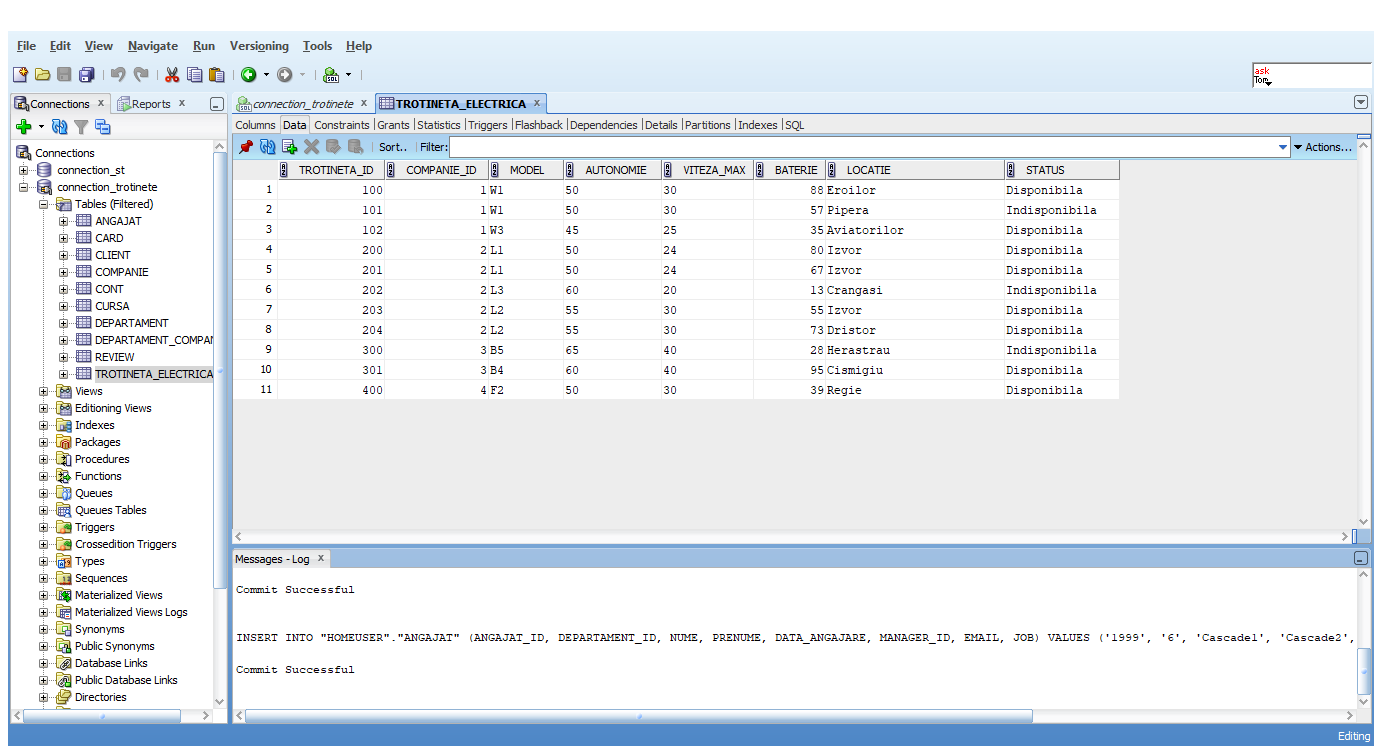
Tabel **Cursa**:



insert into cursa

values (1,'david99',100,null,'Eroilor','Izvor','04-jan-2004',20);

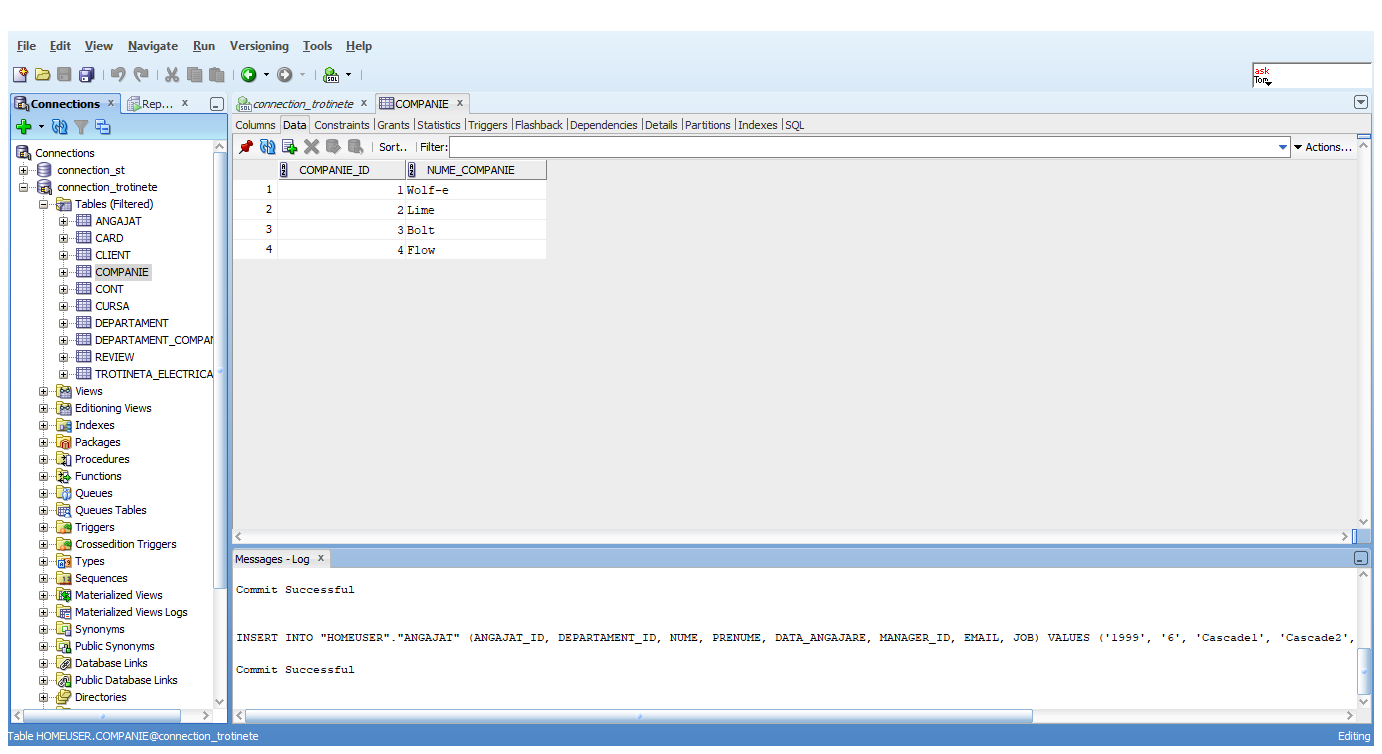
Tabel **Trotineta\_electrica**:



insert into trotineta\_electrica

values (100,1,'A1',50,30,88,'Eroilor',default);

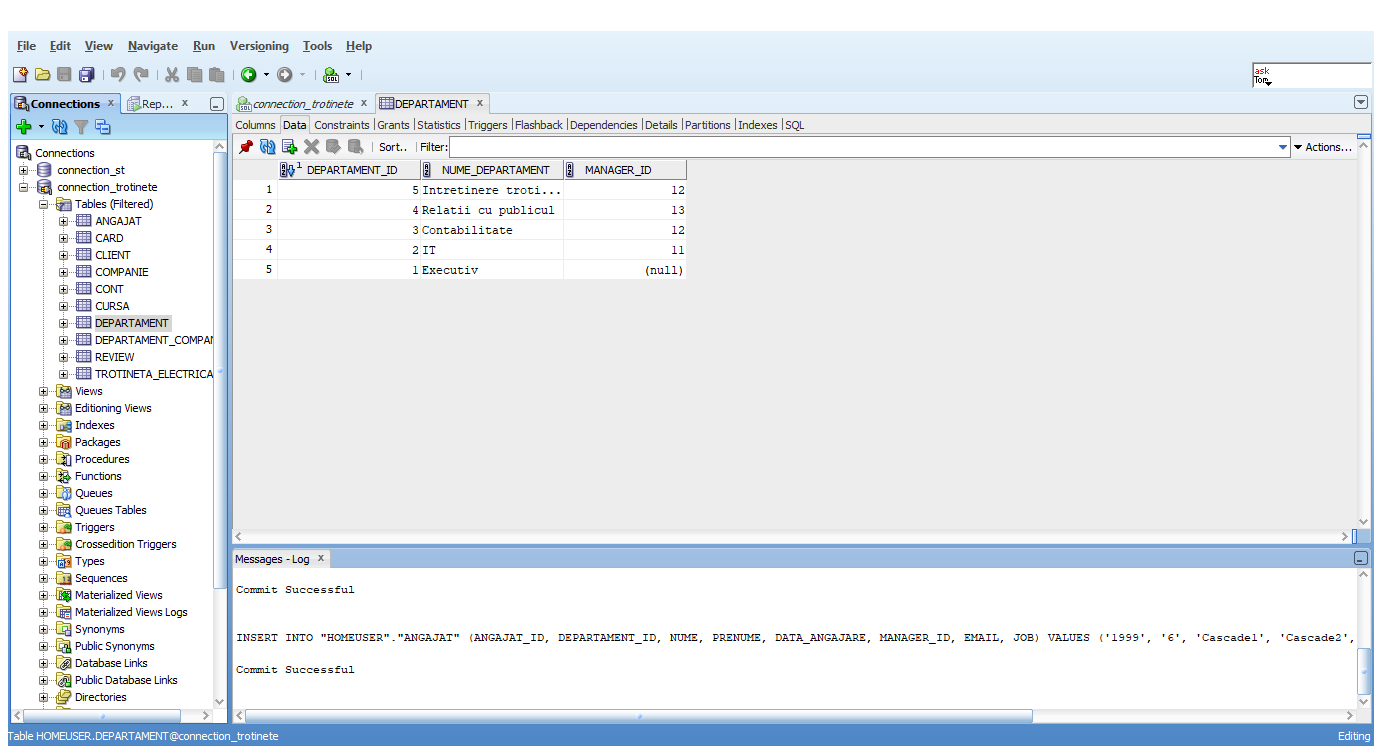
Tabel **Companie**:



insert into companie

values (1,'Wolf-e');

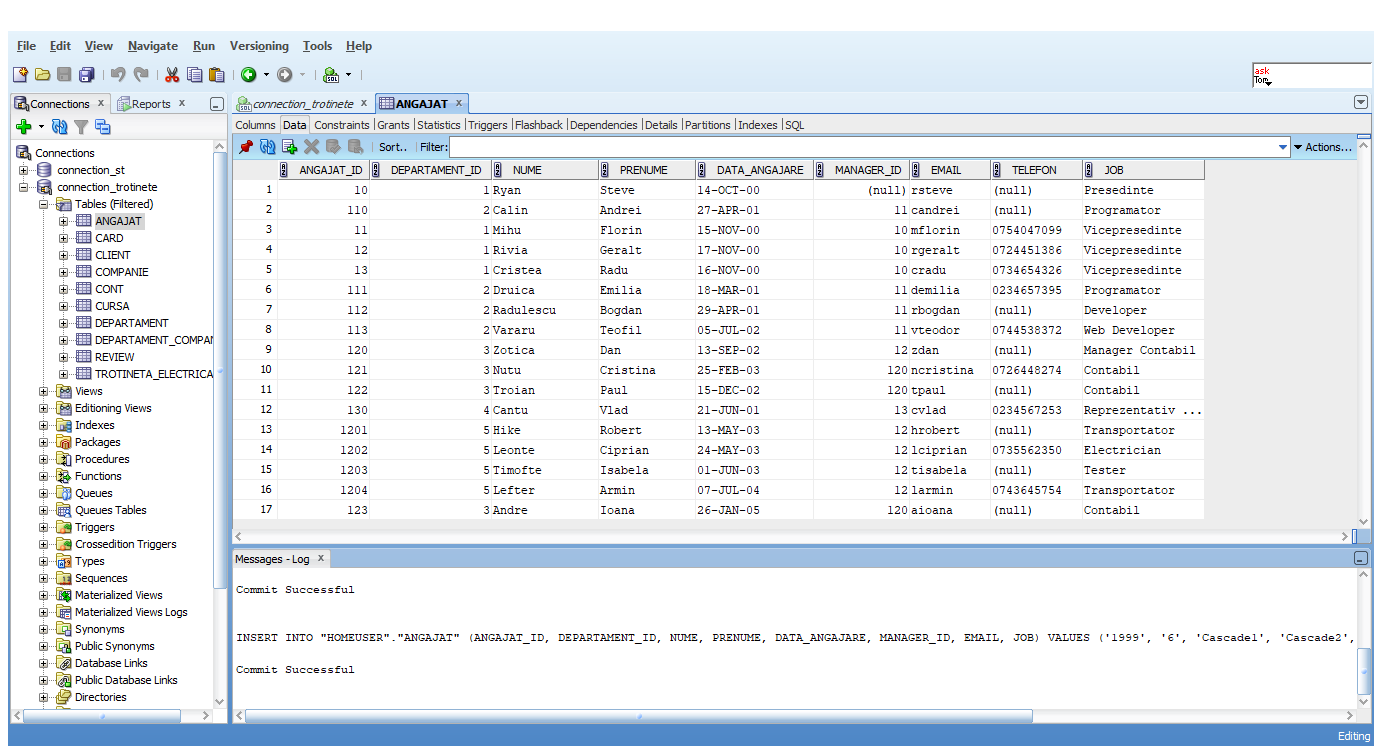
Tabel **Departament**:



insert into departament

values (2,'IT',11);

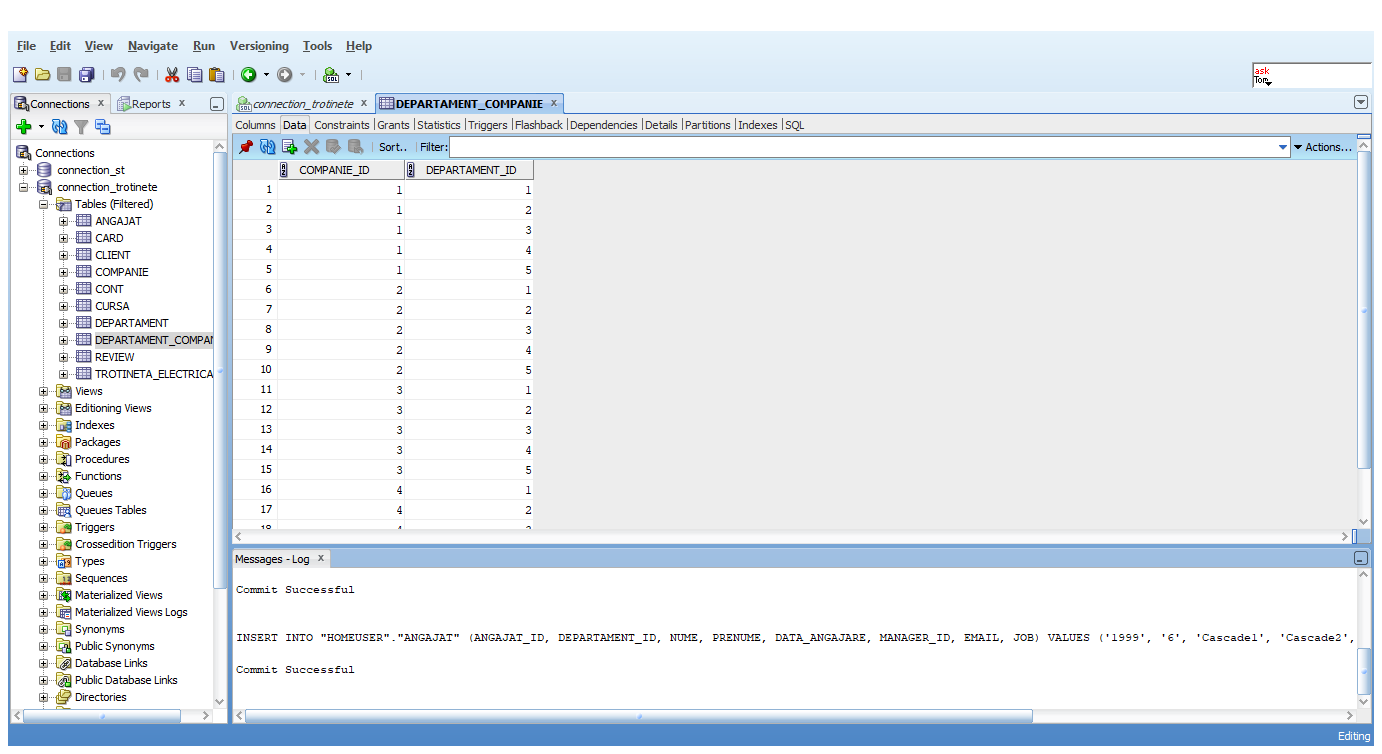
Tabel **Angajat**:



insert into angajat

values (11,1,'Mihu','Florin','15-nov-200' ,10 ,'mflorin' ,'0754047099','Vicepresedinte');

Tabel **Departament\_Companie**:



insert into departament\_companie

values (1,1);