Varianta 1

1. Scrieti un subprogram care returneaza o evidenta cu privire la numarul de pacienti, de fiecare gen, pe care i-a ingrijit fiecare angajat. Subprogramul va returna o singura vaiabila de tiplu lista de obiecte(pentru fiecare angajat se returneaza codul numele functiei detinute, numarul de pacienti de gen feminin, respectiv masculin ingrijiti). Apelati
2. Trigger care la stergerea unui angajat care este manager de specializare, pune in locul acestuia angajatul care a avut cei mai multi pacienti (si codul cel mai mic daca sunt 2 cu aceasta proprietate
3. Subprogram care primeste ca parametru un cod de functie si returneaza prin intermediul unui cursor lista odronata descrescator in functie de salariu, respectiv de cod a angajatilor care nu au tratat pacienti si au functia transmisa.

Varianta 2

AGENTIE_IMOBILIARA (id_agentie, denumire, clasificare)
TRANZACTIONEAZA(cod_agentie, cod_imobil, data_tranzactiei, tip, valoare_comision, pret)
unde tip inseamna vanzare
sau inchiriere
IMOBIL(id_imobil, adresa, cod_proprietar, tip) unde tip inseamna casa, teren sau apartament
PROPRIETAR(id_proprietar, nume)

2. Adaugati o coloana de tip colectie la AGENTIE_IMOBILIARA care sa contina numarul de tranzactii de tip vanzare si numarul de inchirieri efectuate de
fiecare agentie. Presupunand ca acesteau au fost actualizate, modificati clasificarea fiecarei agentii in functie de numarul de preponderent in 'rent', daca
numarul de inchirieri efectuate este mai mare, sau in 'sales' in caz contrar.

3. Creati un subprogram pentru care ii transmiteti un an transmis ca parametru, afisati pentru fiecare agentie : denumirea ei, adresa tuturor imobilelor pe care
le-a tranzactionat si totalul care l-a obtinut din acestea.

4. Creati un trigger care micsoreaza valoarea comisionului cu 25% pentru proprietarii care au facut cel putin 3 tranzactii. Triggerul se va declansa atunci cand
se va insera o noua tranzactie.

5. Creati un subprogram care pentru un nume de proprietar dat ca parametru, intoarce numarul de apartamente care acesta le-a tranzactionat cu firmele care au cele mai
multe de tranzactii incheiate. Tratati exceptiile ce pot aparea.

Varianta 3

Schemele relaţionale ale modelului folosit sunt:
PREZENTARE (cod_pr, data, oras, nume)
SPONSOR (cod_sponsor, nume, info ,tara_origine)
SUSTINE (cod_pr, cod_sp, suma)
VESTIMENTATIE(cod_vestimentatie, denumire, valoare, cod_prezentare)
Obs:

1) Se va lucra in schema exam. Conexiunea în SQL* Plus (sau SQL Developer) se realizează cu:
User: exam
Password: examen
Hostname: 192.168.1.100
Host string (sau Service name): lab223 - unde „” este numarul calculatorului la care sunteti asezati;
2) Soluţiile problemelor vor fi salvate într-un fişier nume_prenume_grupa.txt.
3) Timp de lucru: 90min.
Exerciţii:
1. Adăugaţi constrângerea de cheie externă dintre tabelele PREZENTARE şi VESTIMENTATIE. Implementaţi comportamentul ON DELETE CASCADE cu ajutorul unui trigger. Testaţi trigger-ul. (3p)
2. Să se creeze un subprogram prin care se obţine media valorilor oferite de un sponsor al cărui cod este introdus ca parametru, pentru prezentările în care au existat minim x vestimentatii. Parametrul x va avea valoarea implicita
3. Să se apeleze subprogramul. (3p) 3. Să se creeze un trigger prin care la inserarea unei noi vestimentații pentru o prezentare, suma valorilor vestimentațiilor rămâne mai mică decât suma sponsorizărilor. Altfel, apare o excepție. (3p)

Varianta 4
CLIENT(cod#, nume, prenume, data_nastere)
SERVICIU(cod#, denumire, tip, valoare, cod_notar)
NOTAR(cod, nume, prenume, salariu, data_angajare, procent_comision, oras)
BENEFICIAZA(cod_client#, cod_serviciu#, data#, procent_reducere)
Atributul tip din relația SERVICIU poate avea valorile act sau consultatie.
2. (2p) Creați un subprogram care primeşte ca parametru un cod de notar şi afişează: numele clienților săi și lista denumirilor serviciilor care l-au fost oferite, respectiv numărul de acte eliberate fiecărui client în ultimele 3 luni de acel notar.
3. (2p) Creați un subprogram care primeşte ca parametru un nume de client și intoarce numărul de notari care i-au oferit cele mai ieftine servicii, luând în considerare inclusiv reducerile. Tratați excepțiile.
4. (1,5p) Adăugați o coloană de tip colecție în tabelul NOTAR care pentru fiecare notar va conține lista numerelor sale de telefon. Se presupune că valorile acestei coloane au fost actualizate și că există numere de forma 021/*******. Definiți un bloc PL/SQL care să permită modificarea prefixului numărului de telefon cu o valoare specificată, pentru toți notarii din Bucureşti (ex: prefixul de București se modifică din 021 in 045).
5. (2p) Se presupune că la fiecare 3 consultații de care a beneficiat un client in timpul aceluiaşi an, a patra consultație va fi oferită gratis. Implementați această regulă folosind trigger-i. Actualizarea va avea loc doar în urma operațiilor de adăugare (insert multiplu).

Varianta 5
SPECTACOL(id_spectacol#, denumire, stagiune, cod_sala)
SALA(id sala, denumire, numar_locuri)
BILET_VANDUT(id_bileti, pret, cod_spectacol, data_spectacol, rand, loc)
ACTOR(id_actor, nume, prenume, data_nastere)
JOACA(cod_actor, cod_spectacoll, rol)
1. (2p) Definiți un subprogram stocat care primeşte ca parametru un rol şi întoarce numărul de actori mai mari de 35 de ani care l-au jucat. Trataţi o excepţie la alegere.
2. (3p) Definiți un subprogram stocat care primeşte ca parametru o stagiune şi afişează denumirea spectacolelor din acea stagiune, iar pentru fiecare dintre acestea: lista numelor actorilor care au jucat în spectacolul respectiv şi valoarea totală a biletelor vândute.
3. (2,5p) Definiţi un trigger care să permită achiziționarea unui bilet doar dacă sunt Jocuri libere în sală.

Varianta 6
Schema examen (1.5p):
Implementați o schema (baza de date) formata din 3 (strict) tabele care să aibă relații definite între ele (în soluție vor fi salvate comenzile utilizate la definirea tabelelor). Observații: nu poate fi folosită schema HR și este încurajată definirea unor tabele proprii; nu este obligatorie popularea tabelelor cu date, dar acest lucru poate fi util în rezolvarea problemelor următoare.
II) Întrebări teoretice: a) Ce este o cheie primara, dar o cheie externă? Furnizați câte un exemplu din schema definită mai sus. (0.5p) b) Enumerați câte 2 deosebiri și asemănări dintre un tip de date vector și un tip de date tablou imbricat. (0.5p) c) Ce este un cursor? Care sunt deosebirile dintre un cursor predefinit și un cursor dinamic? (0.5p) d) Enunțați o cerere pe schema aleasă care să poată fi rezolvată cu o funcție, dar să nu poată fi rezolvată cu o procedură. Comentați. (0.5p) e) Enumerați câte 2 deosebiri și asemănări dintre un trigger la nivel de tabel și un trigger la nivel de linie. (0.5p)

III) Probleme (se cere explicit implementarea):
1. Pe schema dată identificați o relație de tipul 1:N (notați în rezolvare alegerea făcută):
a) definiți două tabele care să aibă câte două coloane: cheia primară și o altă coloană de tip vector, respectiv tablou imbricat; (0.5p)
b) cu ajutorul unui subprogram care primește ca parametru o valoarea din partea "one" inserați o linie în primul tabel care să conțină pe prima coloana valoarea data, iar în coloana de tip vector lista valorilor corespunzătoare din cel de al doilea tabel (partea de "many"); (1p)
c) cu ajutorul unui subprogram copiați liniile din primul tabel în cel de al doilea, astfel încât datele să fie sortate crescător în coloana tablou imbricat; (1p)
d) definiți un bloc care utilizează un cursor pentru a afișa conținutul unuia dintre tabelele definite la punctul a) in funcție de o opțiune citită de la tastatura. (0.5p)
2. Simulați cu ajutorul unui trigger constrângerea de cheie primară pe un tabel din schema aleasă. (2p)

Varianta 7
Schemele relaţionale ale modelului folosit sunt:
STATIE(cod_statie, denumire,nr_angajati, cod_companie, capacitate, oras)
ACHIZITIE (cod_st, cod_prod, data_achizitie, cantitate, pret_achizitie)
PRODUS (cod_produs, denumire, pret_vanzare)
COMPANIE (cod, denumire, capital, presedinte)

Exerciţii:
1. Subprogram care primeşte ca parametru un cod de companie şi întoarce lista staţiilor companiei care nu au mai achiziţionat produse în ultimele 10 zile. Apelaţi. (3p)
2. Subprogram care primeşte ca parametru un cod de produs, afişează denumirea şi oraşul staţiilor în care a fost distribuit la un preţ de achiziţie mai mic decât preţul de vânzare. Subprogramul va returna cantitatea totală vandută din produsul dat ca parametru. Trataţi erorile care pot sa apară. Apelaţi. (3p)
3. Să se adauge tabelului statie o coloană stoc care să reprezinte cantitatea totală de produse achiziţionate de fiecare staţie. Actualizaţi această coloană. Să se scrie un trigger care asigură consistenţa acestei coloane. (3p)

Varianta 8
Schemele relaţionale ale modelului folosit sunt:
PERSONAL (id_salariat, nume, prenume, adresa, data_nastere, salariu, id_functie, id_specializare)
PACIENTI (id_pacient, nume, prenume, data_nastere)
TRATEAZA (id_salariat, id_pacient, data_internare, data_externare)
FUNCTII (id_functie, nume_functie, salariu_minim, salariu_maxim)
SPECIALIZARE (id_specializare, nume_specializare, id_manager)

1. Subprogram care primeşte ca parametru un cod de angajat şi returnează lista pacienţilor de care acesta a avut grijă, împreună cu numărul de zile de internare pentru fiecare. Apelaţi.(3p)
2. Subprogram care afişează pentru fiecare funcţie denumirea acesteia împreună cu lista angajaţiilor care au salariul mai mare decât media salarilor colegilor (aceeaşi funcţie) şi care au avut cel puţin doi pacienţi. Trataţi erorile care pot să apară.(3p)
3. Trigger care la ştergerea unui angajat care este manager de specializare, pune în locul acestuia angajatul care a avut cei mai mulţi pacienţi dintre personalul cu specializarea respectivă (se consideră că acesta este unic). (3p)

Varianta 9
Schemele relaționale ale modelului folosit sunt:
TURIST(id_turist,nume.prenume,data_nastere);
AGENTIE(id_agentie denumire.oras);
EXCURSIE(id_excursie,denumire pret, destinatie, durata,cod_agentie.nr_locuri);
ACHIZITIONEAZA(cod_excursie.cod_turist,data_start,data_end.data_achizitie,discount);
Exerciţii:

1. Scrieți un subprogram care primeşte ca parametru codul unei agenţii şi returmează lista ordonată, în funcție de numărul de locuri neocupate, a excursiilor organizate. Apelați. (2p)
2. Să se definească tabelul INFO care să permită stocarea urmatoarelor informații: pentru fiecare agenție (cod) se vor identifica top 3, în funcție de durată, excursii (cod, denumire, numar locun) organizate şi lista turiştilor (cod, nume) care au achiziționat excursii de la fiecare agenție. Dați exemplu de o comandă insert (fără valori null) care adaugă o înregistrare în tabelul INFO. (2.5p)
3. Scrieți un subprogram care populează cu informațiile existente în baza de date tabelul INFO pentru fiecare agenție. (2p)
4. Prin intermediul unui declanşator să se actualizeze automat valoarea reducerilor conform urmatoarelor reguli: 10% pentru excursii cu durata mai mare de 7 zile şi 15% pentru turişti cu vârsta mai mare de 60 de ani. (2.5p)

Varianta 10
Schemele relaționale ale modelului folosit sunt:
FIRMA(cod_f, denumire, data_inf, capital, director)
ANGAJAT(cod_ang, nume, data_nastere, salariu, cod_firma)
LUCREAZA(cod_angajat, id_utilaj, nr_ore, data)
UTILAJ(id_utilaj, denumire, data_achizitie, valoare)
1. Definiți un tabel care să conțină o coloană de tip obiect şi o coloană de tip tablou. Folosiți un bloc anonim care să populeze tabelul definit anterior cu cel puțin 3 linii. (2p)
2. Subprogram care populează tabelul info cu informații (pentru fiecare cod de angajat se rețin codurile utilajelor pe care le-a folosit şi numărul total de ore de muncă pe aceste utilaje). (2p)
3. Trigger care să asigure consistența informațiilor din tabelul info. (2.5p)
4. Procedură care pentru o lună și un cod de firmă, transmise ca parametri, returnează lista angajaţilor care au lucrat mai mult de 10 ore în luna respectivă (2.5p)

Varianta 11
Schemele relaționale ale modelului folosit sunt:
PRODUS (cod_produs, denumire, pret_vanzare);
COMPANIE (cod, denumire, capital, presedinte);
• STATIE(cod_statie, denumire,nr_angajati, cod_companie, capacitate, oras); ;
ACHIZITIE(cod_st,cod_prod,data_achizitie,cantitate, pret_achizitie)
Exerciţii:
1. Scrieți un subprogram care primeşte prin intermediul unui parametru codul unei staţii şi returnează lista ordonată, în funcție de cantitatea totală achiziționată, a produselor comercializate. Apelați. (2p)
2. Să se definească tabelul RAPORT care să permită stocarea următoarelor informații: pentru fiecare companie (cod) se vor identifica top 3, în funcție de numărul de angajaţi, stații (cod, denumire, număr angajați) deținute şi lista produselor (cod, denumire) achiziționate. Adăugați o înregistrare (fără valori null) în tabelul RAPORT. (2.5p)
3. Scrieți un subprogram care populează cu informațiile existente în baza de date tabelul RAPORT pentru fiecare companie. (2p)

4. Prin intermediul unui declanşator să se actualizeze automat prețul de achiziție conform următoarelor reguli: 5% pentru produse cu prețul de vânzare mai mare de 10 şi 7% pentru stații deținute de companii care au un capital mai mic de 500000. (2.5p)