7.01. Se dau relaţiile:

Clienti(idclient, nume, statut)
Bilete(nrbilet, clasa, valoare, sursa, destinația, idclient)
Zboruri(nrzbor, de_la, la, aparat_zbor, nr_locuri, plecare, sosire)
Cupoane(nrbilet, nrzbor, clasa, loc)

Biletul este compus din unul sau mai multe cupoane, fiecare cupon folosind la un zbor între două aeroporturi (*de_la* este aeroportul de unde decolează avionul în data și ora *plecare*, iar *la* este aeroportul unde aterizează avionul în data și ora *sosire*). Numărul de zbor (de exemplu AIF213) constă din două bucăți de informație: prima parte precizează linia de zbor (AIF), iar a doua parte identifică unic ruta de zbor (213). Statut client poate să fie una din valorile {'VIP', 'C'}. Clasa unui bilet și clasa pentru un zbor pot avea doar una din valorile {'Economic', 'Business'}.

Să se scrie următoarele instrucţiuni:
a) creare tabelă pentru relaţia Clienti;
b) creare tabelă pentru relaţia Bilete;
c) creare tabelă pentru relaţia Zboruri;
d) creare tabelă pentru relaţia Cupoane;
e) să se declare cheile primare şi străine;
f) modificare definiţie tabelă Clienti pentru a adăuga atributul *adresa*.

7.02. Să se exprime următoarele constrângeri (la nivel atribut sau tuplă):
a) Atributul *loc* trebuie să fie un număr pozitiv.
b) Dacă clasa unui bilet este 'Economic' atunci valoare nu poate fi mai mare de 500.

7.03. Să se exprime în SQL următoarele interogări:
a) Să se găsească detaliile biletelor pentru care sursa și destinația încep cu aceeași literă în ordinea sursei.
b) Să se găsească pentru *nrbilet* 123 ce *nrzbor*, *clasa* și *loc* au fost folosite.

7.04. Să se exprime în SQL următoarele interogări folosind operatorul JOIN:
a) Să se găsească pentru clientul cu numele 'Popescu Paul' detaliile călătoriilor efectuate în perioada '01-JAN-2018' - '31-MAR-2018'.
b) Să se găsească perechi de zboruri (nrzbor1, nrzbor2) pentru *nrbilet* 123. O pereche este unică în rezultat.

7.05. Să se exprime în SQL fără funcții de agregare următoarele interogări folosind cel puțin o interogare imbricată și operatori de genul EXISTS, IN, ALL, ANY:
a) Să se găsească numele clienților al căror bilet are valoarea cea mai mare între biletele din clasa ’Economic’.
b) Să se găsească destinațiile clienților cu statut VIP cu plecare de pe aeroportul ’Oradea’.

7.06. Să se exprime în SQL următoarele interogări folosind funcţii de agregare:
a) Să se găsească pentru zborurile din perioada ’01-JAN-2018’ - ’31-MAR-2018’ numărul de bilete pentru fiecare clasă.
b) Să se găsească valoarea medie a biletelor pentru fiecare clasă.

7.07. Să se scrie instrucţiunile pentru actualizarea BD:
a) Să se adauge pentru zborul ’AIF213’ ce pleacă de la ‘A’ la ‘B’, folosește aparatul de zbor ‘AIRBUS 310-325’, cu plecare la ora 10:45 și sosire la ora 13:05 în data de ’15-AUG-2018’, cu 100 locuri, biletul 123, clasa ’Economic’, locul 89.
b) Să se șteargă biletele fără cupoane alocate.
c) Să se modifice valoarea biletelor clienților cu statut VIP, pentru a acorda o reducere de 10%.

7.08. Să se definească triggere pentru:
a) A asigura că la adăugarea unui cupon, valoarea atributului loc este între 1 și nr_ locuri al zborului.
b) A împiedica modificarea clasei unui bilet dacă există cupoane alocate.
c) Presupunând vederea:

```
CREATE VIEW BileteVIP AS
SELECT nume as numeclient, nrbilet, sursa, destinatie, clasa, valoare
FROM Clienti NATURAL JOIN Bilete
WHERE statut = ’VIP’;
```

Să se definească un trigger instead-of pentru a permite adăugare prin această vedere.