**VARIANTA 1**

**SQL și Algebra Relațională:**

1. Care este diferența principală între **INNER JOIN** și **LEFT JOIN**?

a) **INNER JOIN** returnează toate combinațiile posibile dintre tabele.  
b) **LEFT JOIN** include toate rândurile din tabelul din stânga, indiferent dacă există corespondență în tabelul din dreapta.  
c) **INNER JOIN** include toate rândurile din ambele tabele, indiferent de corespondențe.  
d) Nu există nicio diferență între ele.

1. Ce face operatorul **EXCEPT** în SQL?  
   a) Returnează intersecția dintre două tabele.  
   b) Returnează diferența dintre două tabele.  
   c) Returnează combinația tuturor rândurilor din două tabele.  
   d) Elimină duplicatele dintr-un tabel.
2. În relația R(A,B,C,D) cu dependențele funcționale B→C și D→A care dintre următoarele este o cheie candidată?  
   a) AB  
   b) BD  
   c) AD  
   d) BC
3. Care dintre următoarele interogări elimină duplicatele din rezultatul său?  
   a) **SELECT ALL**  
   b) **SELECT DISTINCT**  
   c) **SELECT UNIQUE**  
   d) **SELECT WHERE**

**Arbori echilibrați și B-Tree:**

1. Ce este un B-tree de ordin mmm?  
   a) Fiecare nod poate avea cel mult m+1 copii.  
   b) Fiecare nod poate avea cel mult m chei și m+1 copii.  
   c) Fiecare nod trebuie să aibă exact m chei.  
   d) Fiecare nod poate avea cel mult 2m copii.
2. Dacă inserăm cheile **10, 20, 5, 6, 12, 30, 7, 17** într-un B-tree de ordin 3, câte noduri vor fi create?  
   a) 3  
   b) 4  
   c) 5  
   d) 6
3. Care este principalul avantaj al unui B-tree?  
   a) Minimizarea numărului de citiri/scrieri de pe disc.  
   b) Utilizarea eficientă a memoriei RAM.  
   c) Stocarea datelor neordonate.  
   d) Creșterea vitezei de procesare a tranzacțiilor.

**Formele normale:**

1. Care este criteriul principal pentru ca o relație să fie în **2NF**?  
   a) Toate atributele să fie atomice.  
   b) Nu există dependențe tranzitive.  
   c) Nu există dependențe parțiale.  
   d) Toate dependențele funcționale să fie determinate de o supercheie.
2. O relație este în **BCNF** dacă:  
   a) Este în 3NF și fiecare determinant este o cheie candidat.  
   b) Este în 3NF și fiecare atribut este o cheie.  
   c) Este în 2NF și nu există dependențe tranzitive.  
   d) Nu există atribute redundante.
3. În relația R(A,B,C) cu A→B și B→C care este forma normală maximă?  
   a) 1NF  
   b) 2NF  
   c) 3NF  
   d) BCNF

**2. Exerciții practice**

**Arbori echilibrați:**

1. Construiește un **B-tree** de ordinul 3 folosind cheile: **15, 20, 10, 25, 30, 5, 35, 40, 45**. Arată pașii pentru fiecare inserare.
2. Dacă un nod dintr-un B-tree are 4 chei, care este numărul minim de copii pe care îl poate avea nodul?

**Formele normale:**

1. Pentru relația R(A,B,C,D,E) cu dependențele funcționale:
   * A→BC
   * B→D
   * C→E  
     a) Găsește cheile candidate.  
     b) În ce formă normală se află relația?  
     c) Descompune relația în 3NF, dacă este cazul.

Raspunsuri:

1.b, 2.b, 3.b, 4.b, 5.b, 6.b, 7.a, 8.c, 9.a, 10.b

Arbori:

1.

[15, 25, 35]

/ | | \

[5, 10] [20] [30] [40, 45]

2. Într-un B-tree de ordin mmm, un nod cu kkk chei poate avea între k+1k+1k+1 și m+1m+1m+1 copii. Deci, cu 4 chei, nodul poate avea **minimum 5 copii**.

Forme normale:

1.A

2. 2NF

3. R1(A,B,C), R2(B,D), R3(C,E)

**VARIANTA 2**

**SQL și Algebra Relațională:**

1. Care dintre următoarele afirmații este adevărată despre clauza GROUP BY?  
   a) Poate fi utilizată fără funcții agregate.  
   b) Este obligatoriu să fie urmată de o clauză HAVING.  
   c) Funcțiile agregate se aplică asupra grupurilor create de GROUP BY.  
   d) Ordinea coloanelor în GROUP BY nu afectează rezultatul.
2. Ce face comanda SQL de mai jos?

SELECT MAX(salary)

FROM employees

WHERE department\_id IN (

SELECT department\_id

FROM departments

WHERE location\_id = 100

);

a) Returnează cel mai mare salariu din tabelul employees.  
b) Returnează cel mai mare salariu pentru angajații din departamentele aflate în locația 100.  
c) Returnează toate salariile angajaților din locația 100.  
d) Returnează departamentul cu cel mai mare salariu din locația 100.

1. În relația R(A,B,C,D) cu dependențele funcționale A→BC și B→D, ce cheie candidată este validă?  
   a) A  
   b) AB  
   c) AC  
   d) B
2. Ce operator SQL combină rezultatele a două interogări și păstrează duplicatele?  
   a) INTERSECT  
   b) UNION  
   c) UNION ALL  
   d) EXCEPT

**Arbori echilibrați și B-Tree:**

1. Dacă inserăm cheia 15 într-un arbore AVL, iar balanța unui nod devine −2, ce operație este necesară?  
   a) Rotire simplă la stânga.  
   b) Rotire simplă la dreapta.  
   c) Rotire dublă (dreapta-stânga).  
   d) Rotire dublă (stânga-dreapta).
2. Într-un arbore AVL, ce condiție trebuie să respecte fiecare nod pentru ca arborele să fie echilibrat?  
   a) Diferența de înălțime dintre subarborii stânga și dreapta să fie cel mult 2.  
   b) Diferența de înălțime dintre subarborii stânga și dreapta să fie cel mult 1.  
   c) Numărul de noduri din fiecare subarbore trebuie să fie egal.  
   d) Numărul de chei din fiecare nod să fie egal.
3. Care este diferența principală dintre un arbore AVL și un arbore B?  
   a) Arborele AVL este echilibrat pe baza înălțimii, iar arborele B este echilibrat pe baza numărului de chei din noduri.  
   b) Arborele B permite valori duplicate, iar AVL nu.  
   c) Arborele AVL este folosit pentru discuri, iar arborele B pentru memorie RAM.  
   d) Nu există nicio diferență între ele.

**2. Exerciții practice**

**Arbori echilibrați:**

1. Construiește un arbore AVL prin inserarea cheilor: 10, 20, 30, 25, 28, 27. Arată pașii pentru fiecare inserare și ce rotații sunt necesare pentru a menține echilibrul.

**SQL avansat:**

1. **Ai următoarele tabele:**
   * **employees:**

| **employee\_id** | **name** | **salary** | **department\_id** |
| --- | --- | --- | --- |
| **1** | **John** | **5000** | **10** |
| **2** | **Alice** | **6000** | **20** |
| **3** | **Bob** | **4500** | **10** |
| **4** | **Carol** | **7000** | **30** |

* + **departments:**

| **department\_id** | **department\_name** | **location\_id** |
| --- | --- | --- |
| **10** | **IT** | **1** |
| **20** | **HR** | **2** |
| **30** | **Finance** | **1** |

**Task: Scrie o interogare SQL care:**

* **Găsește numele angajaților care lucrează în departamente din locația 1 și au un salariu mai mare decât salariul mediu al tuturor angajaților din departamentele respective.**

**Raspunsuri:**

**1.c, 2.b, 3.a, 4.c, 5.a, 6.b, 7.a**

**ARBORI:**

**25**

**/ \**

**20 30**

**/ /**

**10 28**

**/**

**27**

**2.**

**SELECT e.name**

**FROM employees e**

**JOIN departments d ON e.department\_id = d.department\_id**

**WHERE d.location\_id = 1**

**AND e.salary > (**

**SELECT AVG(e2.salary)**

**FROM employees e2**

**JOIN departments d2 ON e2.department\_id = d2.department\_id**

**WHERE d2.location\_id = 1**

**);**