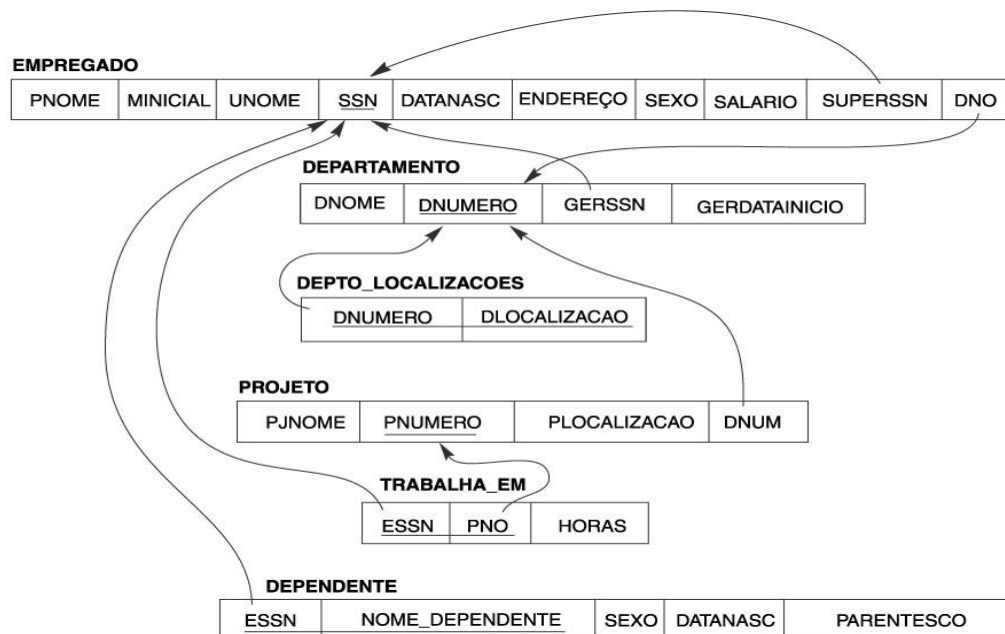


## 2ª Prova

1. (3.0 pontos) Considerando o esquema abaixo, faça a especificação dos seguintes gatilhos em PostgreSQL:
  - a. Gatilho para impedir que um empregado seja alocado em um projeto que não seja do seu departamento.
  - b. Gatilho que incremente em 2% o salário do empregado a cada novo projeto que ele é alocado e que decrescente em 2% a cada projeto que ele é retirado.
  - c. Gatilho para impedir a criação de um projeto caso o departamento tenha quantidade de empregado inferior ao dobro de projetos já existentes para aquele departamento.



2. (2 pontos) Considerando as quatro transações,  $T_1$ ,  $T_2$ ,  $T_3$  e  $T_4$ , e os planos  $S_1$  e  $S_2$  abaixo, construa os grafos de precedência de para  $S_1$  e  $S_2$  e diga se são serializáveis em conflito ou não.

$T_1$ :  $r_1(X)$ ;  $r_1(Z)$ ;  $w_1(X)$ ;  $w_1(Z)$ ;

$T_2$ :  $r_2(Z)$ ;  $r_2(Y)$ ;  $w_2(Z)$ ;  $w_2(Y)$ ;

$T_3$ :  $r_3(X)$ ;  $r_3(Y)$ ;  $w_3(Y)$ ;

$T_4$ :  $r_4(Y)$ ;  $w_4(Y)$ ;  $r_4(Z)$ ;  $w_4(Z)$ ;

$S_1$ :  $r_1(X)$ ;  $r_1(Z)$ ;  $w_1(X)$ ;  $w_1(Z)$ ;  $r_3(X)$ ;  $r_2(Z)$ ;  $r_3(Y)$ ;  $w_3(Y)$ ;  $r_4(Y)$ ;  $w_4(Y)$ ;  $r_2(Y)$ ;  $w_2(Z)$ ;  $r_4(Z)$ ;  $w_2(Y)$ ;  $w_4(Z)$ ;

$S_2$ :  $r_1(X)$ ;  $r_2(Z)$ ;  $r_3(X)$ ;  $r_4(Y)$ ;  $r_1(Z)$ ;  $r_2(Y)$ ;  $r_3(Y)$ ;  $w_1(Z)$ ;  $w_1(X)$ ;  $w_4(Y)$ ;  $w_2(Z)$ ;  $r_4(Z)$ ;  $w_3(Y)$ ;  $w_2(Y)$ ;  $w_4(Z)$ ;

3. (2 pontos) A figura abaixo mostra o *log* de execução de dois escalonamentos das transações  $T_1$ ,  $T_2$ ,  $T_3$ ,  $T_4$  e  $T_5$  até o ponto de queda do sistema. Descreva o processo para recuperação do banco utilizando os protocolos definido para cada *log*. Especifique quais operações serão refeitas ou desfeitas, e qual será o resultado final das variáveis A, B, C e D.

a) Usar UNDO-REDO

```
[start_transaction, T1]
[write, T1, A, 5, 10]
[write, T1, C, 2, 8]
[start_transaction, T2]
[write, T1, D, 20, 25]
[commit, T1]
[write, T2, B, 12, 18]
[write, T2, A, 10, 22]
[start_transaction, T3]
[checkpoint, T2, T3]
[write, T3, D, 25, 30]
[commit, T2]
[write, T3, C, 8, 34]
[start_transaction, T4]
[write, T4, A, 22, 40]
[write, T3, D, 30, 33]
[write, T4, B, 18, 38]
[commit, T3]
[write, T4, A, 40, 55]
[commit, T4]
[start_transaction, T5] ← Queda do Sistema
```

b) Usar NO UNDO-REDO

```
[start_transaction, T1]
[write, T1, D, 20]
[write, T1, C, 40]
[start_transaction, T2]
[write, T2, B, 15]
[checkpoint, T1, T2]
[start_transaction, T3]
[write, T3, B, 12]
[commit, T1]
[write, T2, A, 20]
[commit, T2]
[write, T3, C, 45]
[commit, T3]
[start_transaction, T4]
[write, T4, A, 30]
[start_transaction, T5]
[write, T4, D, 25]
[write, T5, C, 13]
[commit, T4]
[write, T5, B, 18] ← Queda do Sistema
```

4. (3 pontos) Com base nas consultas abaixo faça a representação da árvore algébrica otimizada utilizando as regras de equivalência da álgebra relacional.

Consultas:

- Select e.pnome, d.nome\_dependente  
from empregado e, dependente d  
where e.ssn = d.essn and e.sexo = 'M' and d.parentesco = 'Filha'
- Select e.pnome, e.unome, s.pnome, s.unome  
from empregado e, empregado s  
where e.superssn = s.ssn and e.salario > ( Select AVG(salario) from empregado where superssn is not null)
- Select d.dnome, p.pjnome, e.unome, t.horas  
from departamento d, projeto p, trabalha\_em t, empregado e  
where d.dnumero = p.dnum and p.pnumero = t.pno and e.ssn = t.essn and d.dnome = 'Administração'  
and t.horas > 10