Temps: 2,5 hores

Notes 22 juny Revisió: 28 juny **Cada pregunta s'ha de lliurar en un full separat**

1. **(1 punt)** Donades les base de dades següent:

```
create table professors
(dni char(50),
nomProf char(50) unique,
telefon char(15),
primary key (dni));

create table despatxos
(modul char(5),
numero char(5),
superficie integer not null check(superficie <=25),
primary key (modul,numero));

create table assignacions
(dni char(50),
modul char(5),
numero char(5),
instantInici integer,
instantFi integer,
primary key (dni, modul, numero, instantInici),
foreign key (dni) references professors,
foreign key (modul,numero) references despatxos,
check (instantInici < instantFi));
```

Considereu el següent fragment de codi Java/JDBC. Implementeu el cos del **while** per tal que s'incrementi en 5 unitats la superfície dels despatxos de cadascun dels mòduls obtinguts en la variable varModul. Definiu també els Statements o PreparedStatements necessaris on creieu convenient. Cal que el programa tregui un missatge d'excepció si la superfície d'algun despatx passa a ser superior a 25. En cas que no hi hagi cap excepció el programa indicarà el número de despatxos modificats de cada mòdul.

```
try {
 /* Tenim ja una connexió c establerta amb la base de dades */
 /* Tenim un variable in que permet llegir el que escriu l'usuari */

 System.out.print("Escriu el nom d'un mòdul: ");
 String varModul = in.nextLine();

 while (!varModul.equals("acabar")) {

     //codi a implementar

     System.out.print("Escriu el nom d'un mòdul: ");
     varModul = in.nextLine();
     }
 c.commit();
 }
catch (ClassNotFoundException ce) {
        System.out.println ("Error al carregar el driver");}
catch (SQLException se) {
        System.out.println ("Excepcio: "+ se.getMessage());}
```

Recordeu que podeu usar els mètodes/codis d'excepció de JDBC estudiats a classe:

Statements

- Statement createStatement();
- ResultSet executeQuery(String sql);
- int executeUpdate(String sql);

PreparedStatement

- PreparedStatement prepareStatement(String sql);
- ResultSet executeQuery();
- int executeUpdate();
- void setXXX(int posicioParametre, XXX valor);

ResultSet

- boolean next();
- XXX getXXX(String nomColumna)

SQLException

- // 23502 -not_null_violation, 23503 -foreign_key_violation
- // 23505 -unique_violation, 23514 -check_violation
- String getSQLState();

SOLUCIÓ

```
try {
    /* Tenim ja una connexió c establerta amb la base de dades */

    System.out.println("Escriu el nom d'un mòdul: ");
    String varModul = System.console().readLine();

    PreparedStatement ps = c.prepareStatement("update despatxos d "+
                                    "set superficie = superficie + 5 "+
                                    "where d.modul = ?");


    while (!varModul.equals("acabar")) {

        //codi a implementar
        ps.setString(1,varModul);
        int numFilesActualitzades = ps.executeUpdate();
        System.out.println (varModul+"  "+numFilesActualitzades);
        System.out.println("Escriu el nom d'un mòdul: ");
        varModul = System.console().readLine();
        }
    c.commit();
    }
   catch (ClassNotFoundException ce) {
             System.out.println ("Error al carregar el driver");}
   catch (SQLException se) {
            if(se.getSQLState().equals("23514"))
                  System.out.println("La superficie no pot ser més gran que 25");
            else
                  System.out.println ("Excepcio: "+ se.getMessage());}
```

2. (1 punt) Suposeu la base de dades següent:

```
empleats1 (numEmp1, nomEmp1, ciutatEmp1);
empleats2 (numEmp2, nomEmp2, ciutatEmp2);
```

Suposeu que es vol implementar la restricció següent mitjançant disparadors:

Els valors de l'atribut `ciutatEmp1` de la taula `empleats1` han d'estar inclosos en els valors de l'atribut `ciutatEmp2` de la taula `empleats2`

La idea és llançar una excepció en cas que s'intenti executar una sentència que violi la restricció.

2.1 Digueu quins són els esdeveniments rellevants (taula/es i esdeveniment/s).

empleats1	Insert	
empleats1	Update	ciutatEmp1
empleats2	Delete	
empleats2	Update	ciutatEmpl2

2.2 Digueu i justifiqueu de quin tipus han de ser el disparadors (ROW / STATEMENT, BEFORE/AFTER).

Tots els disparadors han de ser before, for each row.

Before perquè com que es tracta de que salti una excepció quan es violi una restricció d'integritat, com més aviat es descarti que la restricció es viola millor, per tal d'evitar fer feina innecessària.

For each row perquè és millor una solució incremental. Suposant que a cada taula hi ha una gran quantitat d'empleats, i que els esdeveniments afecten a poques files, una solució incremental sempre comportarà menys accessos a la base de dades, perquè només requereix fer comprovacions per a les tuples que s'han inserit/esborrat/modificat.

2.3 Implementeu un dels disparadors que hagueu identificat per a la taula empleats1. En cas de que no n'hi hagi cap justifiqueu perquè. Podeu fer servir la plantilla següent.

```
CREATE FUNCTION comprovarCiutat() RETURNS trigger AS $$
    BEGIN
     //implementació
    END;
$$LANGUAGE plpgsql;

CREATE TRIGGER ex3_3
    BEFORE/AFTER <esdeveniment>
    FOR EACH ROW/STATEMENT EXECUTE PROCEDURE comprovarCiutat();
```

```
CREATE FUNCTION comprovarCiutEmpl1() RETURNS trigger AS $$
    BEGIN
     IF  not exists(select* from empleats2
         where  ciutatEmp2=new. ciutatEmp1) THEN
         RAISE EXCEPTION 'Error en inserir o modificar empleats1';
     END IF;
     RETURN NEW;
    END;
$$LANGUAGE plpgsql;

CREATE TRIGGER ex3_3
    BEFORE INSERT OR UPDATE of ciutatEmp1 ON empleats1
    FOR EACH ROW EXECUTE PROCEDURE comprovarCiutEmpl1();
```

3. **(2 punt)** Considereu la base de dades de l'exercici 1 amb el contingut següent:

Professors	dni	nomProf	telefon	sou
	111	Joana	97743121	10000
	222	Martí	93818903	3000
	333	Lourdes	97261201	50000

Despatxos	modul	numero	superficie
	C6	101	13
	Omega	300	8
	Omega	301	8

Assignacions	dni	modul	numero	instantIni	instantFi
	333	C6	101	5	9
	111	C6	101	3	NULL
	222	Omega	300	10	NULL

Considereu també el procediment emmagatzemat següent:

```
CREATE TYPE despatx AS (d_modul char(5), d_numero char(5));

CREATE FUNCTION assignaProfessor(dni_p char(50), modul_d char(5),
                                 numero_d char(5), inici_a integer)
            RETURNS SETOF despatx AS $$
DECLARE desp despatx;
BEGIN
  IF (NOT EXISTS(SELECT * FROM Professors WHERE dni=dni_p)
       OR NOT EXISTS(SELECT * FROM Despatxos
                     WHERE modul=modul_d AND numero=numero_d)) THEN
       RAISE EXCEPTION 'El professor o el despatx no existeix';
     ELSIF (EXISTS (SELECT * FROM Assignacions a
                    WHERE a.dni=dni_p AND a.instantFi IS NULL)) THEN
          UPDATE Assignacions SET instantFi = inici_a - 1
          WHERE dni=dni_p AND instantFi IS NULL;
   END IF;
   INSERT INTO Assignacions VALUES (dni_p,modul_d,numero_d,inici_a,NULL);
   FOR desp IN SELECT d.modul,d.numero
                  FROM Professors p NATURAL JOIN Assignacions a
                                    NATURAL JOIN Despatxos d
               WHERE a.instantFi IS NULL
               GROUP BY d.modul, d.numero
               HAVING COUNT(*) >= 2
    LOOP
      RETURN NEXT desp;
    END LOOP;
   EXCEPTION
       WHEN raise_exception THEN raise exception '%',sqlerrm;
 END;
 $$LANGUAGE plpgsql;
```

3.1 Indiqueu i justifiqueu quin és l'estat de la base de dades i el valor de retorn després de l'execució de la consulta: `SELECT * FROM assignaProfessor('111','Omega','300',15);`

Les taules Professors i Despatxos no es modifiquen, el contingut de la taula Assignacions és:

Dni	modul	numero	instantIni	instantFi
333	C6	101	5	9
111	C6	101	3	14
222	Omega	300	10	NULL
111	Omega	300	15	NULL

El procediment ha finalitzat totes les assignacions actives del professor amb dni_p (posant com a instant fi el d'inici menys 1), i n'ha creat una de nova per al despatx donat com a paràmetre. L'execució de la consulta retorna:

d_modul	d_numero
Omega	300

els quals corresponen als despatxos compartits (més d'un professor assignat) amb assignacions actives (instantFi amb valor NULL).

3.2 Digueu quins són els criteris de qualitat que no es compleixen en la implementació del procediment donat i com caldria canviar el codi per tal que es complíssin.

- Accessos innecessari a la BD.
  - El primer IF amb els seus SELECTs no cal. Per tant els SELECTs són innecessaris perquè l'excepció es pot capturar mitjançant la violació de la condició de foreign key que saltarà en fer el INSERT de l'assignació nova.
  - El ELEIF amb els seu SELECT no calen. El UPDATE ja només modifica les assignacions amb instantFi NULL si és que n'hi ha alguna.
- TAULES+JOINs innecessàries
  - En el SELECT que serveix per buscar el resultat que ha de donar el procediment, les taules Professors i Desspatxos no són necessàries.

4. **(3 punts)** Transaccions:

4.1 Suposeu ara l'horari següent.

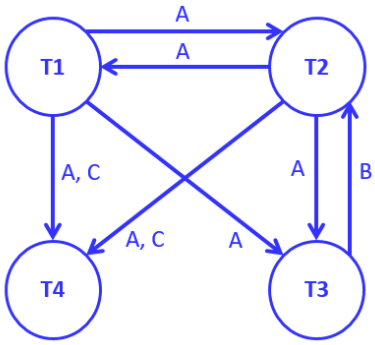
a) Doneu el graf de precedències per aquest horari.

b) En cas que hi hagi alguna interferència en l'horari resultant digueu quina/es són, les transaccions i el/s gràNul/s implicat/s.

c) És serialitzable? En cas afirmatiu doneu el horari serial equivalent. En cas negatiu perquè no.

	T1	T2	T3	T4
10			R(B)	
20		RU(A)		
30	RU(A)			
40		W(A)		
50		R(C)		
60		R(F)		
70	W(A)			
80				R(A)
90		RU(B)		
100	R(C)			
110				RU(C)
120		W(B)		
130			R(A)	
140				W(C)
150	commit			
160		commit		
170			commit	
180				commit

a) Graf de precedències.



b) Hi ha una actualització perduda entre T1 i T2 i un anàlisi inconsistent entre T2 i T3
c) No és serialitzable perquè hi ha interferències.

4.2 En cas que les transaccions treballin en nivell d'aïllament READ COMMITTED. Doneu l'horari un cop aplicades les reserves corresponents al nivell d'aïllament. En l'horari heu d'incloure, a més de les operacions que executen les transaccions (R, RU, W, COMMIT), les operacions de petició i alliberament de reserves (LOCK, UNLOCK), i l'ordre d'execució de totes aquestes operacions. Quan més d'una transacció espera per un mateix gràNul, considereu que la política de la cua és FIFO (First In, First Out).

T1	T2	T3	T4
		L(B,S)	
		R(B)	
		U(B)	
	L(A,X)		
	RU(A)		
L(A,X)			
	W(A)		
	L(C,S)		
	R(C)		
	U(C)		
	L(F,S)		
	R(F)		
	U(F)		
			L(A,S)
	L(B,X)		
	RU(B)		
		L(A,S)	
	W(B)		
	C+U(A,B)		
RU(A)			
W(A)			
L(C,S)			
R(C)			
U(C)			
C+U(A)			
			R(A)
			U(A)
			L(C,X)
			RU(C)
		R(A)	
		U(A)	
			W(C)
		C	
			C+U(C)

4.3 Considereu ara un SGBD sense cap mecanisme de control de concurrència, on el gràNul és la fila. I considereu l'horari següent que té una interferència d'anàlisi inconsistent. Expliqueu la interferència d'anàlisi inconsistent en base a l'horari i al contingut de la base de dades donat en l'exercici 3.

	T1	T2
1	select * from despatxos where modul='Omega' and numero=300;	
2		update despatxos set superficie = superficie + 8 where modul='Omega' and numero=300;
3		select * from despatxos where modul='Omega' and numero=301;
4	update despatxos set superficie = superficie + 5 where modul='Omega' and numero=301;	
5	Commit	
6		Commit

La interferència d'anàlisi inconsistent té a veure amb la visió que dues transaccionstenen d'un conjunt de dades. En aquest horari el conjunt de dades són les files 'Omega', 300 i 'Omega', 301.

En aquest horari:

- T1 llegeix les files amb valors 'Omega', 300, 8 i 'Omega', 301, 8.
- T2 llegeix les files amb valors 'Omega', 300, 8 i 'Omega', 301, 8

L'horari no és correcte perquè no dóna el mateix resultat que cap dels dos horaris serial.

En cas de l'horari seria T1;T2:

- T1 llegeix les files amb valors 'Omega', 300, 8 i 'Omega', 301, 8
- T2 llegeix les files amb valors 'Omega', 300, 8 i 'Omega', 301, 13

En cas de l'horari seria T2;T1:

- T2 llegeix les files amb valors 'Omega', 300, 8 i 'Omega', 301, 8
- T1 llegeix les files amb valors 'Omega', 300, 16 i 'Omega', 301, 8

5. **(3 punt)** Considereu la taula R(a,b,c,d). Aquesta taula té 100.000 files, i el factor de bloqueig, de les pàgines de dades on s'emmagatzemen aquestes files, és de 10 files per pàgina.

Considereu la consulta següent.

```
SELECT *
FROM R
WHERE b=5 and c >=50
```

Se sap que hi ha 70 files que compleixen la condició b=5, 100 files la condició c>=50 i només 1 fila les dues condicions alhora.

IMPORTANT: En tots els apartats cal detallar tots els càlculs que feu per calcular els costos.

5.1 Suposant que <u>només podem definir un únic índex, arbre B+, per un únic atribut</u>, i que aquest índex tindrà una ocupació del 80% i un ordre d=50, Quin tipus d´índex (agrupat o no agrupat) escolliríeu i per quin atribut hauria d'estar definit per tal de fer la consulta el més eficient possible? Justifiqueu les respostes en base els costos de totes les opcions possibles.

SOLUCIÓ:

Les opcions possibles són definir un únic índex agrupat o no agrupat pels atributs b o c, i aplicar la resta de condicions a mida que es van obtenint les files de les pàgines de dades. Qualsevol altre índex no augmentaria l'eficiència de la consulta.

Índex agrupat R.b: Cost = h +D = 3 +7 = 10

Valors per node = $\lceil 2d*0,8 \rceil = \lceil 2*50*0,8 \rceil$ = 80 valors / node

h = $\lceil \log_{81} 100000 \rceil$ = 3

D= $\lceil$ card(R.b=5) / 10 $\rceil = \lceil 70 / 10 \rceil = 7$

Índex no agrupat R.b: Cost = h + F + card(R.b=5) = 3 +0 + 70 = 73

h = $\lceil \log_{81} 100000 \rceil$ = 3

F= $\lceil 70 / 80 \rceil$ - 1= 0

card(R.b=5) = 70

Índex agrupat R.c: Cost = h +D = 3 +10 = 13

Valors per node = $\lceil 2d*0,8 \rceil = \lceil 2*50*0,8 \rceil$ = 80 valors / node

h = $\lceil \log_{81} 100000 \rceil$ = 3

D= $\lceil$ card(R.c>=50) / 10 $\rceil = \lceil 100 / 10 \rceil = 10$

Índex no agrupat R.c: Cost = h + F + card(R.c > 50) = 3 +1 + 100 = 104

h = $\lceil \log_{81} 100000 \rceil$ = 3

F= $\lceil 100/ 80 \rceil$ - 1= 1

card(R.c > =50) = 100

La millor opció serà definir un índex agrupat per l'atribut R.b, amb un cost de 10 accessos a disc.

5.2 Suposant ara que en lloc d'un únic índex per un únic atribut, en podem definir 2 també per un únic atribut i del mateix tipus que abans (una ocupació del 80% i un ordre d=50) quin seria el cost fent servir l'estratègia de intersecció de RIDs?

Cost índex per R.b = h + F = 3

Cost índex per R.c = h + F = 4

Cost per acedir a dades = Card(R.b=5 and R.c >50) = 1

El cost seria 8 accesos a disc

5.3 Suposant ara que podem definit un índex no agrupat multiatribut pels atributs b, c. Quin seria el cost fent servir aquest índex en cas de tenir ordre d=25 i ocupació 80%?

el cost seria: h +F +Card(R.b=5 and R.c >=50) = 4 + 0 +1 = 5

Valors per node = $\lceil 2d*0,8 \rceil = \lceil 2*25*0,8 \rceil$ = 40 valors / node

h = $\lceil \log_{41} 100000 \rceil$ = 4

F = F= $\lceil 1/ 80 \rceil$ - 1= 0