Examen de
Bases de datos y sistemas de información

1) (0,6 puntos) Dados los esquemas de relación R(A,B) y S(A,C), se pide expresar las siguientes expresiones del cálculo relacional de dominios en álgebra relacional y cálculo relacional de tuplas:
   a) $\{<A> | \exists B(<A,B> \in R \wedge B=2)\}$
   b) $\{<A,B,C> | (<A,B> \in R \wedge <A,C> \in S)\}$

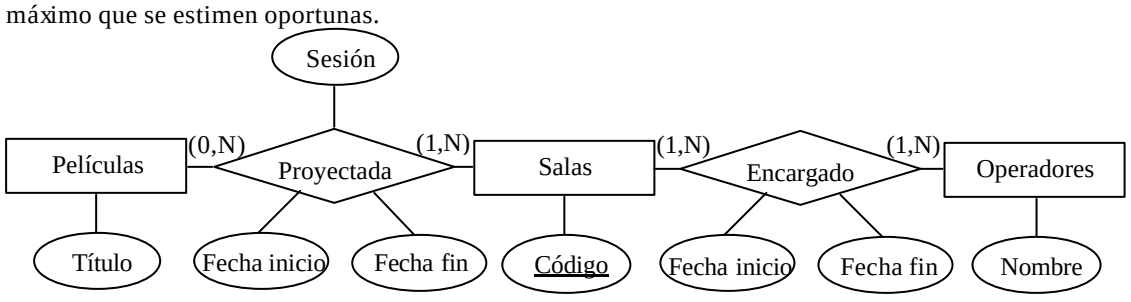
   R:

a) $\{<A> | \exists B(<A,B> \in R \wedge B=2\}$
AR: $p_A(s_{B=2}(R))$
CRT: $\{s | \exists t \in R (t[B] = 5 \wedge t[A] = s[A])\}$

b) $\{<A,B,C> | (<A,B> \in R \wedge <A,C> \in S\}$
AR: $R \rtimes S$
CRT: $\{t | \exists p \in R(\exists q \in S (t[A] = p[A] \wedge p[A] = q[A] \wedge t[B] = p[B] \wedge t[C] = q[C]))\}$

2) (2,5 puntos) Un multicine con varias salas desea contar con una base de datos. Cada sala está identificada por un código y en cada una de ellas hay varias sesiones, cada una de las cuales se identifica por su hora de inicio. Cada sala debe estar atendida por un operador (no siempre el mismo necesariamente), identificado por su nombre, que se encarga de la proyección, siendo posible que el mismo operador atienda varias salas. Las películas, identificadas por su título, se proyectan durante un intervalo de fechas, pero es posible que alguna no se haya asignado aún a una sala. En una sala sólo se proyecta una película diferente al día. Las películas pueden reponerse. Los operadores están asignados a las salas por intervalos de fechas. Se pide:
   a) Construir el modelo entidad-relación explicando el significado de los conjuntos de entidades y relaciones, imponiendo y explicando las restricciones de clave, de dominio y de cardinalidad mínimo - máximo que se estimen oportunas.
   b) Traducir el modelo conceptual obtenido al modelo lógico sin considerar las operaciones y expresando las restricciones de integridad referencial y de participación total en notación algebraica.
   c) Imponer la siguiente restricción de integridad en el modelo lógico: las sesiones de cada proyección de película deben tomar valores de un conjunto predefinido.
   d) Plantear las siguientes consultas SQL:
      i) Listado de todas las películas proyectadas por un operador.
      ii) Listado de las películas que todavía no se han asignado a ninguna sala.
      iii) Número de días de proyección de cada sala.

   a) Construir el modelo entidad-relación explicando el significado de los conjuntos de entidades y relaciones, imponiendo y explicando las restricciones de clave, de dominio y de cardinalidad mínimo - máximo que se estimen oportunas.



a1) Significado de los conjuntos de entidades y relaciones
- Conjuntos de entidades:
  - Películas: Películas proyectadas en el multicine identificadas por su nombre.
  - Salas: Salas del multicine identificadas por su código.

    - Operadores: Operadores del multicine identificados por su nombre.
- Conjuntos de relaciones:
    - Proyectada: Películas × Salas
      Identifica la sala y sesión en la que se proyecta cada película durante un intervalo de fechas.
    - Encargado: Operadores × Salas
      Identifica el operador encargado de cada sala durante un intervalo de fechas.

a2) Restricciones de clave y de dominio

- Conjuntos de entidades:
    - Películas(Título: String). Las películas están identificadas por su título.
    - Salas(Código: String). Las salas están identificadas por su código.
    - Operadores(Nombre: String). Los operadores están identificados por su nombre.
- Conjuntos de relaciones:
    - Proyectada(Sesión: String, Fecha inicio: Date, Fecha fin: Date)
    - Encargado(Fecha inicio: Date, Fecha fin: Date)

a3) Restricciones de cardinalidad mínimo -máximo

Proyectada:

- Películas - (0,N) - Proyectada - Salas: Es posible que una película aún no se proyecte en ninguna sala (mínimo 0). Una película se puede proyectar en varias salas (máximo N).
- Películas - Proyectada - (1,N) - Salas: En cada sala se ha proyectado al menos una película (mínimo 1). En cada sala se pueden proyectar varias películas (en diferentes periodos de tiempo) (máximo N).

Encargado:

- Operadores - (1,N) - Encargado - Salas: Cada operador está encargado al menos de una sala (mínimo 1). El mismo operador puede estar encargado de varias salas (máximo N).
- Operadores - Encargado - (1,N) - Salas: Cada sala debe tener un operador asignado (mínimo 1). Cada sala puede tener asignados varios operadores (en diferentes periodos de tiempo) (máximo N).

b) Traducir el modelo conceptual obtenido al modelo lógico sin considerar las operaciones y expresando las restricciones de integridad referencial y de participación total en notación algebraica.
No es necesaria una fase de reestructuración (generalizaciones, mezclas o divisiones).

Modelo relacional:

- Entidades:
    - Películas(Título)
    - Salas(Código)
    - Operadores(Nombre)
- Relaciones:
    - Proyectada(Película, Sala, Fecha inicio, Fecha fin, Sesión)
      Es necesario que Fecha inicio sea también parte de la clave para admitir los reestrenos.
    - Encargado(Operador, Sala, Fecha inicio, Fecha fin)
      Es necesario que Fecha inicio sea también parte de la clave para admitir que un operador sea encargado de la misma sala en diferentes momentos.

Restricciones de integridad referencial:

Con respecto a Proyectada:

- $\Pi_{Película}(\text{Pr}\,oyectada) \subseteq \Pi_{Título}(Películas)$
- $\Pi_{Sala}(\text{Pr}\,oyectada) \subseteq \Pi_{Código}(Salas)$

Con respecto a Encargado:

- $\Pi_{Operador}(Enc\,\text{arg}\,ado) \subseteq \Pi_{Nombre}(Operadores)$
- $\Pi_{Sala}(Enc\,\text{arg}\,ado) \subseteq \Pi_{Código}(Salas)$

Restricciones de participación total:

Con respecto a Proyectada:

- $\Pi_{Sala}(\text{Pr}\,oyectada) = \Pi_{Código}(Salas)$

Con respecto a Suministran:

- $\Pi_{Operador}(Enc\,\text{arg}\,ado) = \Pi_{Nombre}(Operadores)$

- $\Pi_{Sala}(Enc\arg ado) = \Pi_{Código}(Salas)$

c) Imponer la siguiente restricción de integridad en el modelo lógico: las sesiones de cada proyección de película deben tomar valores de un conjunto predefinido.

Se crea una nueva relación Sesiones(Sala: String, Sesión: Time) que contenga todas las posibles sesiones de cada sala y se añade la siguiente relación de integridad referencial:

$$\Pi_{Sala,Sesión}(\Pr oyectada) \subseteq \Pi_{Sala,Sesión}(Sesiones)$$

d) Plantear las siguientes consultas SQL:

i) Listado de todas las películas proyectadas por un operador.

```
PARAMETERS NombreOperador;
SELECT Película, Operador
FROM Proyectada, Encargado
WHERE NombreOperador=Operador
  AND Proyectada.Sala=Encargado.Sala
  AND Encargado.Fecha inicio
      BETWEEN Proyectada."Fecha inicio" AND Proyectada."Fecha fin"
  OR Encargado.Fecha fin
      BETWEEN Proyectada."Fecha inicio" AND Proyectada."Fecha fin";
```

ii) Listado de las películas que todavía no se han asignado a ninguna sala.

```
SELECT Título
FROM Películas NATURAL LEFT OUTER JOIN Proyectada
WHERE Proyectada.Película IS NULL;
```

Otra alternativa, con operaciones de conjunto:

```
(SELECT Título FROM Películas) MINUS
(SELECT Película AS Título FROM Proyectada);
```

Otra alternativa más, con EXIST:

```
SELECT Título
FROM Películas
WHERE NOT EXIST (SELECT *
                 FROM Proyectada
                 WHERE Proyectada.Película = Película.Título);
```

Una última alternativa, con IN:

```
SELECT Título
FROM Películas
WHERE Título NOT IN (SELECT Película
                     FROM Proyectada);
```

iii) Número de días de proyección de cada sala.

```
SELECT Sala, SUM(DAYS("Fecha fin" - "Fecha inicio"))
FROM Proyectada
GROUP BY Sala;
```

3) (1,8 puntos) Dada la relación R(A,B,C,D) y las dependencias funcionales AD $\rightarrow$ C, B $\rightarrow$ C y C $\rightarrow$ D, se pide:

a) Descomponer R para obtener un conjunto de esquemas en FNBC.
b) ¿La transformación ha preservado las dependencias funcionales?
c) Descomponer R para obtener un conjunto de esquemas en 3FN.

R:
a) Descomponer R para obtener un conjunto de esquemas en FNBC.
Hay que comprobar en primer lugar si no está ya en FNBC (todos los antecedentes de las D.F. deben ser superclave):
{AD}+ = {A,C,D} No es superclave
{B}+ = {B,C,D} No es superclave
{C}+ = {C,D} No es superclave

Se descompone R aplicando el algoritmo de descomposición a FNBC:

```
D={R}
while Q Î D, t.q. D no está en FNBC
 {encontrar X->Y de Q que viole FNBC
  reemplazar Q por Q-Y y XÈY}
```

AD → C
Q-Y= S(A,B,D)
X∪Y=T(A,C,D)

Para comprobar si S(A,B,D) está en FNBC:
- Determinar el cierre de F (F+).
- Comprobar que para cada X → Y ∈ F+, X es superclave.

{A}+ = {A}
{B}+ = {B,C,D} C ∉ S, Se deduce {B → D} en este esquema, {B} no es superclave,

Se descompone S(A,B,D) con respecto a {B → D}

S1 = {A,B}
S2 = {B,D}
Se ha terminado en los dos casos porque son relaciones de dos atributos.

Para comprobar si T(A,C,D) está en FNBC:
- Determinar el cierre de F (F+).
- Comprobar que para cada X → Y ∈ F+, X es superclave.

{A}+ = {A}
{C}+ = {C,D}, Se deduce {C → D} en este esquema, {C} no es superclave

Se descompone T con respecto a {C → D}

T1 = {A,C}
T2 = {C,D}
Se ha terminado en los dos casos porque son relaciones de dos atributos.

b) ¿La transformación ha preservado las dependencias funcionales?
Para comprobar si preserva las dependencias funcionales se podría aplicar el algoritmo:

```
  for each D.F. X ® Y
1.  Z:={X}
    while cambios en Z do
      for i:=1 to k do
2.      Z := Z È ((ZÇRi)+ÇRi)
```

Si $Y \subseteq Z \Rightarrow X \rightarrow Y \in G^+$
Si hay algún $X \rightarrow Y$ t.q. Y no es subcjto. de Z, significa que $X \rightarrow Y$ no $\in G^+$

y, por tanto, no se conservan las D.F.

Sin embargo, como AD → C es una D.F. que relaciona tres atributos y en la descomposición sólo aparecen relaciones de dos, se sabe que esta D.F. no se preservará.

c) Descomponer R para obtener un conjunto de esquemas en 3FN.

En primer lugar se comprueba si se encuentra ya en 3FN:
Para toda X → Y, o bien X es superclave o Y contiene algún atributo que pertenece a una clave candidata.

No hay claves candidatas de un atributo, como se ha visto en el apartado a) al calcular {A}+, {B}+ y {C}+ y siendo {D}+ = {D}.
De dos atributos:
{AB}+ = {A,B,C,D} Clave candidata
{AC}+ = {A,C,D}
{AD}+ = {A,C,D}
{BC}+ = {B,C,D}
{BD}+ = {B,C,D}
{CD}+ = {C,D}

Para AD → C: AD no es superclave y C no pertenece a ninguna clave candidata. Por lo tanto, R NO ESTÁ EN 3FN.

Se aplica el algoritmo de descomposición en 3FN:

D= {}
1. Encontrar un recubrimiento mínimo T de F
2. for each X ® Y Î T, crear en D un esquema Ri con {X È{A1}È...È{Ak}}
si Ri ⊄ Rj ∈ D, dadas las D.F. X ®{A1}, ..., X ® {Ak}
3. Si ninguno de los esquemas contiene una clave de R, se crea uno nuevo.

1. Para ver que F es minimal:
1. Por la aplicación del lema 2 (una dependencia funcional $X \to \{B_1,\ldots,B_n\}$ se puede dividir en n dependencias funcionales $X \to \{B_1\},\ldots, X \to \{B_n\}$) se obtiene un conjunto T equivalente a F de manera que todas las partes derecha de sus dependencias funcionales tienen un solo atributo.
2. Para cada dependencia funcional $X \to \{B\} \in T$, $X = \{A_1,\ldots,A_n\}$ se examina cada atributo A_i: si la eliminación de A_i de X no tiene efecto sobre F^+, se elimina A_i de X.

3. Para cada dependencia funcional restante, si su eliminación no tiene efecto sobre F^+, se elimina de T.
Para AD → C:
1. Se cumple
2. Se elimina A:
Ahora S'={D → C, B → C, C → D}. Para ver si AD → C se deduce de S', se calcula {AD}+={ADC}, y de aquí se infiere que AD → C. (Se podría calcular completamente el cierre de S y compararlo con el de S').
Para B → C:
Si se elimina esta D.F., S'+ ≠ S+, ya que B no aparecería en S'.
Para C → D:
Si se elimina esta D.F., a simple vista S'+ ≠ S+, pero hay que comprobarlo. Se ve simplemente dado que en S', {C}+={C}, por lo que nunca se puede deducir C → D de S'.
Por tanto, *F* es minimal.

2.D → C: {C,D}
2.B → C: {B,C}
2.C → D: {C,D} No se incluye porque ya lo tenemos de antes

R1(C,D)
R2(B,C)

Vemos si CD es clave de R:
{CD}+={C,D}
No lo es y, por tanto, hay que crear un nuevo esquema que contenga la clave primaria de R:
R3(A,B)

4) (1,2 puntos) Dada la relación Padre(P,H) que determina los pares <P,H> tales que P es padre de H, se pide:
   a) Determinar la clave primaria del esquema.
   b) Plantear una consulta SQL que obtenga los hijos de Paco.
   c) Plantear una consulta SQL que obtenga los nietos de Paco.
   d) Plantear una consulta SQL que obtenga los biznietos de Paco.
   e) ¿Es posible encontrar una consulta que pueda averiguar todos los descendientes de Paco?

R:
a) Determinar la clave del esquema.
Padre(P,H)
b) Plantear una consulta SQL que obtenga los hijos de Paco.
```
SELECT H
FROM Padre
WHERE P='Paco';
```
c) Plantear una consulta SQL que obtenga los nietos de Paco.
```
SELECT H
FROM Padre
WHERE P IN
(SELECT H
 FROM Padre
 WHERE P='Paco');
```
d) Plantear una consulta SQL que obtenga los biznietos de Paco.
```
SELECT H
FROM Padre
WHERE P IN
(SELECT H
 FROM Padre
 WHERE P IN
  (SELECT H
   FROM Padre
   WHERE P='Paco'));
```
e) ¿Es posible encontrar una consulta que pueda averiguar todos los descendientes de Paco?
No es posible dar una solución totalmente general con SQL sin recursividad, dado que hay que establecer caminos en un árbol de longitud indeterminada, lo cual sólo es posible realizando el producto cartesiano $padre^N$ (Padre × ... × Padre, N veces) siendo N dependiente del problema.
Una solución aproximada es calcular los hijos, unirlos con los nietos, unirlos con los biznietos, y así sucesivamente hasta una generación máxima.
Si las consultas anteriores se han almacenado como Hijos, Nietos y Biznietos, la siguiente consulta calcularía todos los descendientes de Paco hasta sus biznietos:
```
SELECT H FROM Hijos
UNION
SELECT H FROM Nietos
UNION
SELECT H FROM Biznietos;
```

5) (1,2 puntos) Expresar la reunión natural externa por la izquierda de dos tablas P(A,B) y Q(A,C) en Oracle con
   a) la sintaxis clásica de Oracle
   b) la sintaxis del estándar SQL-92
   c) asumiendo que no se dispone de ningún tipo de reunión externa.

a) la sintaxis clásica de Oracle
```
SELECT *
FROM P, Q
```

```
WHERE P.A(+)=Q.A;
```

b) la sintaxis del estándar SQL-92

```
SELECT *
FROM A NATURAL LEFT OUTER JOIN B;
```

Otra alternativa menos concisa:

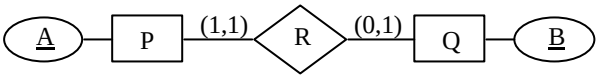
```
SELECT *
FROM A LEFT OUTER JOIN B
ON P.A=Q.A;
```

c) asumiendo que no se dispone de ningún tipo de reunión externa.

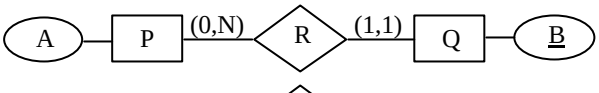
```
SELECT P.A AS PA, P.B AS PB, Q.A AS QA, Q.C AS QC
FROM P, Q
WHERE P.A=Q.A
UNION
SELECT P.A AS PA, P.B AS PB, NULL AS QA, NULL AS QC
FROM P
WHERE NOT EXISTS (SELECT Q.A
                  FROM Q
                  WHERE Q.A=P.A);
```

6) (0,9 puntos) Traducir los siguientes esquemas conceptuales al modelo relacional minimizando el número de tablas e imponiendo las restricciones de integridad (claves primarias, candidatas e integridad referencial) que se estimen oportunas para conservar el mayor grado de integridad. No se permite la aparición de valores nulos en ninguna de las tablas resultantes. Las claves de las tablas que representen a los conjuntos de relaciones deben ser primarias.

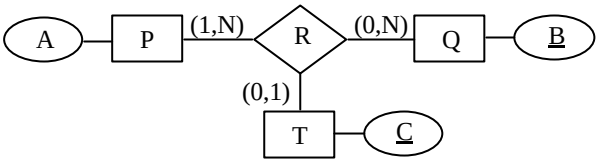
a)



b)



c)



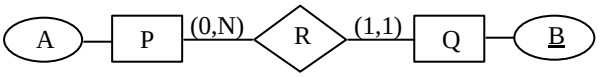
R:

a)

R(A, B), con clave candidata {B}
Q(B)
No es necesario que P aparezca en la traducción porque todas sus entidades deben estar en R, al contrario de lo que ocurre con Q (no todas las entidades deben aparecer en R y, por lo tanto, es necesario una representación explícita de Q).

$$\Pi_B(R) \subseteq \Pi_B(Q)$$

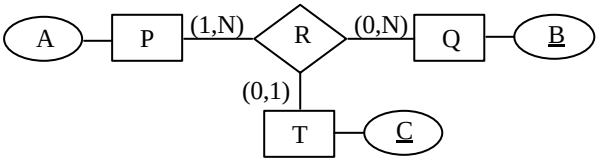
b)



R(A, B). La clave primaria es B debido a la cardinalidad máxima de B en R. No hay claves candidatas.
P(A).
No es necesario que Q aparezca en la traducción porque todas sus entidades deben estar en R, al contrario de lo que ocurre con P (no todas las entidades deben aparecer en R y, por lo tanto, es necesario una representación explícita de P).

$$\Pi_A(R) \subseteq \Pi_A(P)$$

c)



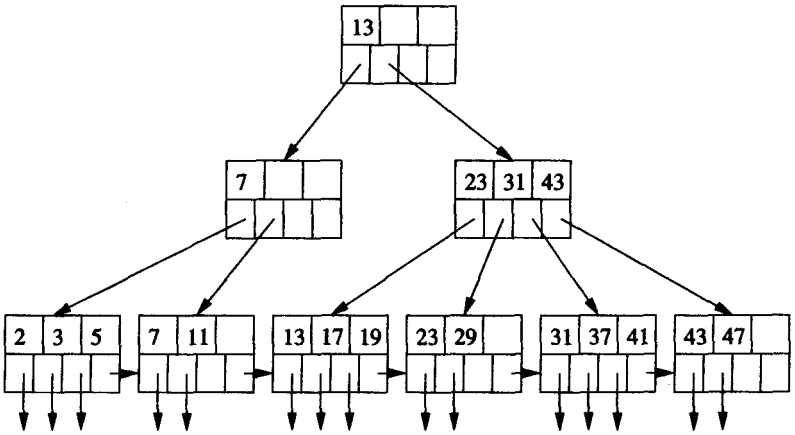
R(A, B, C). No es necesario incluir ni a A ni a B como parte de la clave porque C tiene una participación máxima de 1 en la relación (sus valores no se repetirán en R).
Q(B)
T(C)
No es necesario que P aparezca en la traducción porque todas sus entidades deben estar en R, al contrario de lo que ocurre con Q y T (no todas sus entidades deben aparecer en R y, por lo tanto, es necesario una representación explícita de Q y T).

$$\Pi_B(R) \subseteq \Pi_B(Q)$$

$$\Pi_C(R) \subseteq \Pi_C(T)$$

7) (0,9 puntos) Muéstrese el resultado de insertar en un árbol B+ todos los números primos comprendidos entre el 2 y el 47, en el orden: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47 y con n=4.

R:



8) (0,9 puntos. **Sólo plan 98**) Determínese si la siguiente planificación es legal en:
   a) Protocolo de dos fases.
   b) Protocolo de marcas temporales, asumiendo que un bloqueo conlleva una lectura y un desbloqueo una escritura.

Tiempo	T0	T1
1	LOCK A	
2		LOCK B
3		UNLOCK B
4	LOCK B	
5	UNLOCK A	
6	UNLOCK B	

R:
a) Protocolo de dos fases.
Obviamente lo es porque en ambas transacciones todos los bloqueos preceden a los desbloqueos.

b) Protocolo de marcas temporales.
No es legal en este protocolo porque en el paso 4 la marca temporal en escritura de B es 1.

9) (0,9 puntos. **Sólo plan 91**) Dada la tabla Gastos(Año, Mes, Importe, Concepto), se pide:
   a) Determinar, sin usar consultas anidadas, el gasto mensual del año 2002 sólo de los meses con un gasto superior a 4000 €.
   b) ¿Por qué la siguiente consulta es incorrecta? ¿Cómo habría que reescribirla para obtener la lista de expresiones que aparece en la instrucción SELECT?

```
SELECT Mes, Año, MAX(Importe);
FROM Gastos;
```

R:
a) Determinar, sin usar consultas anidadas, el gasto mensual del año 2002 sólo de aquellos meses con un gasto superior a 4000 €.

```
SELECT Mes, SUM(Importe)
FROM Gastos
WHERE Año=2002
GROUP BY Mes
HAVING SUM(Importe)>4000;
```

b) ¿Por qué la siguiente consulta es incorrecta? ¿Cómo habría que reescribirla para obtener la lista de expresiones que aparece en la instrucción SELECT?
Hay dos errores. En primer lugar, sobra la coma después de la lista de expresiones de la sentencia SELECT. En segundo lugar, no es posible tener un campo que no forme parte de una función de agregación en la lista de expresiones de la instrucción SELECT y que no forme parte de la lista de agrupación.

La consulta correcta sería:

```
SELECT Mes, Año, MAX(Importe)
FROM Gastos
GROUP BY Mes, Año;
```