Ficheros y Bases de Datos
Curso 2009-10
Ingeniería Técnica de Informática
Primer Parcial. 1-Junio-2010

Nombre:_____

Se debe entregar esta hoja
2 horas

1) (3,5 puntos) A partir de la información sobre la BD que se describe más abajo, se pide:
a) (2 puntos) El esquema entidad / relación, incluyendo atributos, claves y restricciones de cardinalidad y/o participación.
b) (1 puntos) Pasar al modelo relacional optimizado.
c) (0,5 puntos) Represente las reglas de integridad referencial resultantes.

Se desea diseñar una base de datos para la gestión de las prácticas con coche en una autoescuela y poder así optimizar los recursos existentes.

La autoescuela tiene bastantes sucursales repartidas por distintas ciudades.
De cada sucursal se suele conocer un Nº de Sucursal, su dirección, ciudad, teléfono y persona de contacto. Las sucursales disponen de coches de práctica.

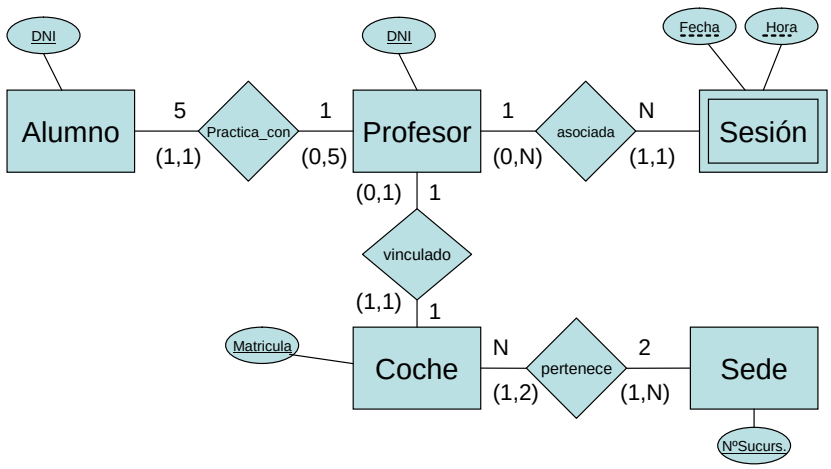
Un coche de prácticas pertenece siempre a la flota de una o dos sucursales, dependiendo de las necesidades. Cada coche está siempre asignado a un único profesor. De los coches se registra su matrícula, modelo, año y daños.

Cada profesor tendrá asignado como máximo un coche. También tiene asignado un número variable de sesiones de prácticas, aunque en ocasiones puede no tener ninguna, por lo que se dedicará a dar teoría en la autoescuela y por tanto no tendrá asignado un coche. Todas las sesiones son asignadas por defecto a un profesor.

De cada profesor se quiere registrar su DNI, nombre, apellidos y las fechas y los correspondientes horarios de las sesiones prácticas asignadas.

Además, cada profesor puede tener hasta un máximo de 5 alumnos, mientras que cada alumno tiene siempre un único profesor de prácticas. La ficha del alumno contiene su DNI, nombre, apellidos, importe pagado, profesor.

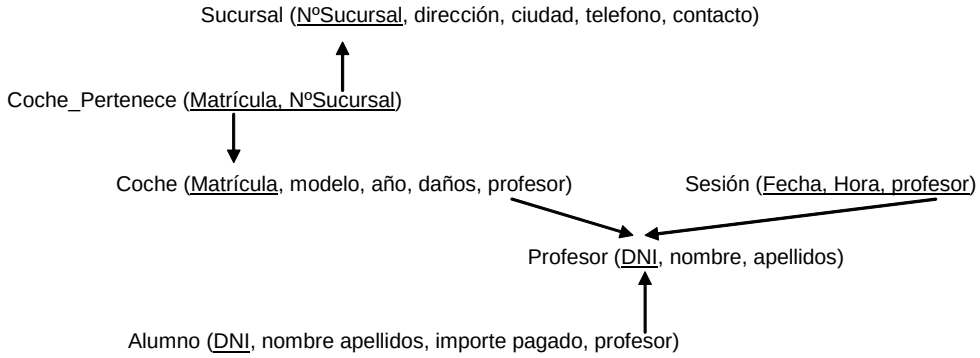
Solución:



b)

Sucursal (NºSucursal, dirección, ciudad, telefono, contacto)
Coche (Matrícula, modelo, año, daños,profesor)
Coche_Pertenece (Matrícula, NºSucursal)
Profesor (DNI, nombre, apellidos)
Sesión (Fecha, Hora, profesor)
Alumno (DNI, nombre apellidos, importe pagado, profesor)

c)



Nota: Algunas restricciones que no conserva dicho modelo relacional son:

- Cada coche está siempre asignado a un único profesor (la restricción de participación total)
- Los profesores tienen como máximo 5 alumnos
- Los coches pertenecen como mucho a dos sedes

2) (4 puntos) Para organizar los prestamos entre países europeos para financiar la deuda pública, se ha diseñado el siguiente modelo relacional:

Préstamo (numPrestamo, paísPrestador, paísPrestado, fecha, cuantía)
País(nombre, PIB, numHabitantes)

Realizar las siguientes consultas SQL:

a) (0,5 puntos) Listar sin duplicados todos los países cuyo nombre empiece por 'A' de más de un millón de habitantes que han prestado dinero a Grecia.

```
SELECT DISTINCT PaisPrestador
FROM préstamo,  País
WHERE  paisPrestador = Nombre
      AND numHabitantes >1000000
      AND nombre LIKE 'A%'
      AND paisPrestado = 'Grecia'
```

b) (0,5 puntos) Mostar los países que no han pedido dinero prestado o no han prestado nada ordenados por nombre. Usar subconsultas correlacionadas.

```
SELECT nombre
FROM País P
WHERE NOT EXISTS (SELECT *
                  FROM Préstamo
```

```
                  WHERE paísPrestador = P.nombre
                  )
      OR
      NOT EXISTS (SELECT *
                  FROM Préstamo
                  WHERE paísPrestado = P.nombre
                  )
ORDER BY nombre
```

c) (0,5 puntos) Mostar los países que no han pedido dinero prestado o no han prestado nada ordenados por nombre. Usar subconsultas no correlacionada.

```
SELECT nombre
FROM País P
WHERE nombre NOT IN (SELECT paísPrestador
                  FROM Préstamo
                  )
      OR
      nombre NOT IN (SELECT paísPrestado
                  FROM Préstamo
                  )
ORDER BY nombre
```

d) (0,5 puntos) Mostar los países que no han pedido dinero prestado o no han prestado nada ordenados por nombre. No usar subconsulta.

```
SELECT nombre
FROM País P LEFT OUTER JOIN Préstamo
      ON nombre = paísPrestador
WHERE fecha IS NULL

UNION

SELECT nombre
FROM País P LEFT OUTER JOIN Préstamo
      ON nombre = paísPrestado
WHERE fecha IS NULL

ORDER BY nombre
```

e) (0,5 puntos) Mostrar todos los países que no han pedido nada prestado con el número de prestamos que han concedido y la cuantía total prestada. Si no han prestado nada, dichos totales deben ser cero

```
SELECT  paísPrestador, count(*) AS numPrestamos, SUM(cuantía)
FROM Préstamo
WHERE paísPrestador NOT IN (SELECT paísPrestado
                              FROM Prestamo)
GROUP BY paisPrestador

UNION ALL

SELECT nombre, 0, 0
FROM país
WHERE nombre NOT IN (SELECT  paísPrestador
                  FROM Préstamo)
      AND nombre NOT IN (SELECT paísPrestado
                  FROM Prestamo)
```

f) (0,5 puntos) ¿Qué país ha prestado más dinero? Indicar cuanto ha prestado en total dicho pais?

```
SELECT paisPrestador, SUM(cuantía)
FROM Préstamo
GROUP BY paisPrestador
HAVING SUM(cuantía) >= ALL (SELECT SUM(cuantía)
                            FROM Préstamo
                            GROUP BY paisPrestador)
```

g) (0,5 puntos) Mostrar todos los países que han prestado y han recibido préstamos, con su déficit (diferencia entre el total recibido y total que han prestado).

```
SELECT Prestadores.paísPrestador, Prestados.totalRecibido –
                                  Prestadores.totalPrestado AS déficit
FROM (SELECT paisPrestador, SUM(cuantía) AS totalPrestado
      FROM Prestamo
      GROUP BY paisPrestador) AS Prestadores
INNER JOIN
      (SELECT paisPrestado, SUM(cuantía) AS totalRecibido
      FROM Prestamo
      GROUP BY paisPrestado) AS Prestados
ON Prestadores.paisPrestador = Prestados.paisPrestado
```

h) (0,5 puntos) Para salvar al euro, todos los países con un PIB superior a mil millones de euros condonan las deudas de Grecia de todos los prestamos anteriores a 2010. Actualizar la base de datos

```
DELETE FROM Préstamo
WHERE paísPrestado = 'Grecia'
  AND fecha < '01.01.2010'
  AND paísPrestador IN (SELECT nombre
                        FROM País
                        WHERE PIB > 1000000000)
```

3) (2,5 puntos) Dado la relación R(A,B,C) y su conjunto de dependencias funcionales S={AB→C,C →A}

Responder las siguientes preguntas justificando las respuestas.

a) (0,5 punto) Comprueba si estas dependencias funcionales son minimales.
b) (0,5 puntos) Calcular las claves candidatas de S
c) (0,5 puntos) Determinar si la relación R está en 3FN.
d) (0,5 puntos) Determinar si la relación R está en FNBC.
e) (0,5 puntos) La descomposición en R_1(B, C) y R_2(A, C), ¿preserva la propiedad de reunión no aditiva?. ¿preserva las dependencias funcionales?

Solución:

a) Aplicamos el algoritmo visto en clase:

1. G:=S, G={AB→ C, C→A}
2. *Separar* las dependencias funcionales con más de un atributo en la parte derecha (por ejemplo, X→YZ) por las correspondientes dependencias *simples*, X→Y, X→Z (Axioma de descomposición). En este caso no se produce cambio en G.
3. Eliminación de atributos redundantes

Evaluamos la dependencia funcional AB→C.

¿Se encuentra B→C en G^+?

Calculamos $\{B\}^+ = B$

Cuestionamos si ¿$C \in \{B\}^+$?. La respuesta es no y por tanto no podemos eliminar el atributo B en AB→C.

¿Se encuentra A→C en G^+?

Calculamos $\{A\}^+ = A$

Cuestionamos si ¿$C \in \{A\}^+$?. La respuesta es no y por tanto no podemos eliminar el atributo A en AB→C.

Por ahora G no se reescribe: G={AB→C, C→A}

4. Eliminación de dependencias funcionales redundantes

¿ $C \in \{AB\}_{G-\{AB->C\}}^+$? No, porque $\{AB\}_{G-\{AB->C\}}^+ = AB$. Según el algoritmo, para calcular aquí $\{AB\}^+$ no se tiene en cuenta la dependencia AB→C en G.

¿ $A \in \{C\}_{G-\{C->A\}}^+$? No, porque $\{C\}_{G-\{C->A\}}^+ = C$. Según el algoritmo, para calcular aquí $\{C\}^+$ no se tiene en cuenta la dependencia C→A en G.

Según estos resultados no existen DFs redundantes que eliminar.

G no se reescribe y por tanto nos queda $G_{min} = S_{min}$={AB→C, C→A} como el conjunto mínimo de dependencias funcionales equivalente.

b) Comprobamos que AB→R, calculando el cierre del conjunto de atributos AB:

$\{AB\}^+ = AB \cup C$, $ABC \supseteq R$; AB es superclave para R

$\{A\}^+ = A$, *A* no es superclave para R

$\{B\}^+ = B$, *B* no es superclave para R

Como la superclave AB no contiene otra superclave, podemos afirmar que AB es clave candidata.

Otra clave candidata es BC.

c) Según la definición, una relación R está en normalizada en 3FN si:

Para cualquier dependencia funcional X→Y no trivial en S^+,

- X es superclave en R, es decir X→ R, o bien
- Y forma parte de una clave candidata en R.

Aprovechando el punto a) estudiaremos las dependencias funcionales X→ Y en S_{min}, ya que S_{min} es un subconjunto válido de S^+. Por tanto,

AB→C y C→A están en S^+

Vemos que la dependencia funcional AB→ C no es trivial; es decir, {C} no es un subconjunto de {AB}. Además, el antecedente X, es decir {AB}, es superclave en R, por tanto cumple la primera opción.

Vemos que la dependencia funcional C→ A no es trivial; {A} no es un subconjunto de {C}. Además, el consecuente Y, es decir {A}, forma parte de la una clave candidata en R; A $\subseteq$ {A, B}.

Podemos afirmar que la relación R está normalizada según 3FN.

Otra forma es aplicar el algoritmo de descomposición de 3FN y ver que R no cambia, pues está en 3FN.

d) Según la definición, una relación R está en normalizada en FNBC si:

Para cualquier dependencia funcional X→Y no trivial en S^+,

X es superclave en R, es decir X→ R.

Vimos en el apartado anterior que la dependencia funcional AB→ C no es trivial; es decir, {C} no es un subconjunto de {AB}. Además, el antecedente {AB} es superclave en R, por tanto cumple con la normalización FNBC. Pero vemos que, aunque la dependencia funcional C→ A no es trivial; el antecedente {C}, NO es superclave en R, por tanto NO cumple con la forma normal FNBC.

e) Para ver si esta descomposición $R_1(B,C)$ y $R_2(A,C)$ preserva la propiedad de reunión no aditiva, podemos comprobar la propiedad vista en clase:

$((R_1 \cap R_2) \rightarrow (R_1 - R_2)) \in S^+$ o bien
$((R_1 \cap R_2) \rightarrow (R_2 - R_1)) \in S^+$

Si tomamos este último caso, tenemos que:

$((BC \cap AC) \rightarrow (AC - BC)) = C \rightarrow A \in S^+$

Podemos afirmar que la descomposición propuesta sí preserva la propiedad de reunión no aditiva.

Nota: Otra forma de demostrarlo es ver que la descomposición dada es la salida del algoritmo que descompone en FNBC, que preserva la propiedad de reunión no aditiva

En esta descomposición se ve que no se preserva la dependencia funcional AB→C, pues ni R_1 ni R_2 contienen todos los atributos {A, B, C}