FUNDAMENTOS DE BASES DE DATOS
Solución Examen Julio 2014

La duración del examen es de 3 horas.
Presentar la resolución del examen:

- Con las hojas numeradas y escritas de un solo lado.
- Con la cantidad de hojas entregadas en la primer hoja.
- Con cédula de identidad y nombre en cada hoja.
- Escrita a lápiz y en forma prolija.
- Comenzando cada ejercicio en una nueva hoja

Ejercicio 1 (25 puntos).

Un gobierno departamental desea implementar un sistema de información para la gestión del transporte público.

Los ómnibus son propiedad de las empresas de transporte colectivo de pasajeros, las cuales se identifican por su nombre. Además, de cada empresa se conoce su dirección (calle y número) y una lista de teléfonos. De los ómnibus, que se identifican por su matrícula, se conoce su marca, año de empadronamiento y cantidad de asientos. Los ómnibus pueden tener o no asiento para guarda.

Los recorridos se identifican por un nombre y representan una secuencia de paradas, donde para cada una de ellas se sabe su posición relativa dentro del recorrido (EJ: la parada 0 es el origen, etc.). Algunos recorridos pueden estar contenidos dentro de otros e interesa modelar esto. Cada parada se identifica por sus coordenadas (latitud y longitud) y tiene un nombre.

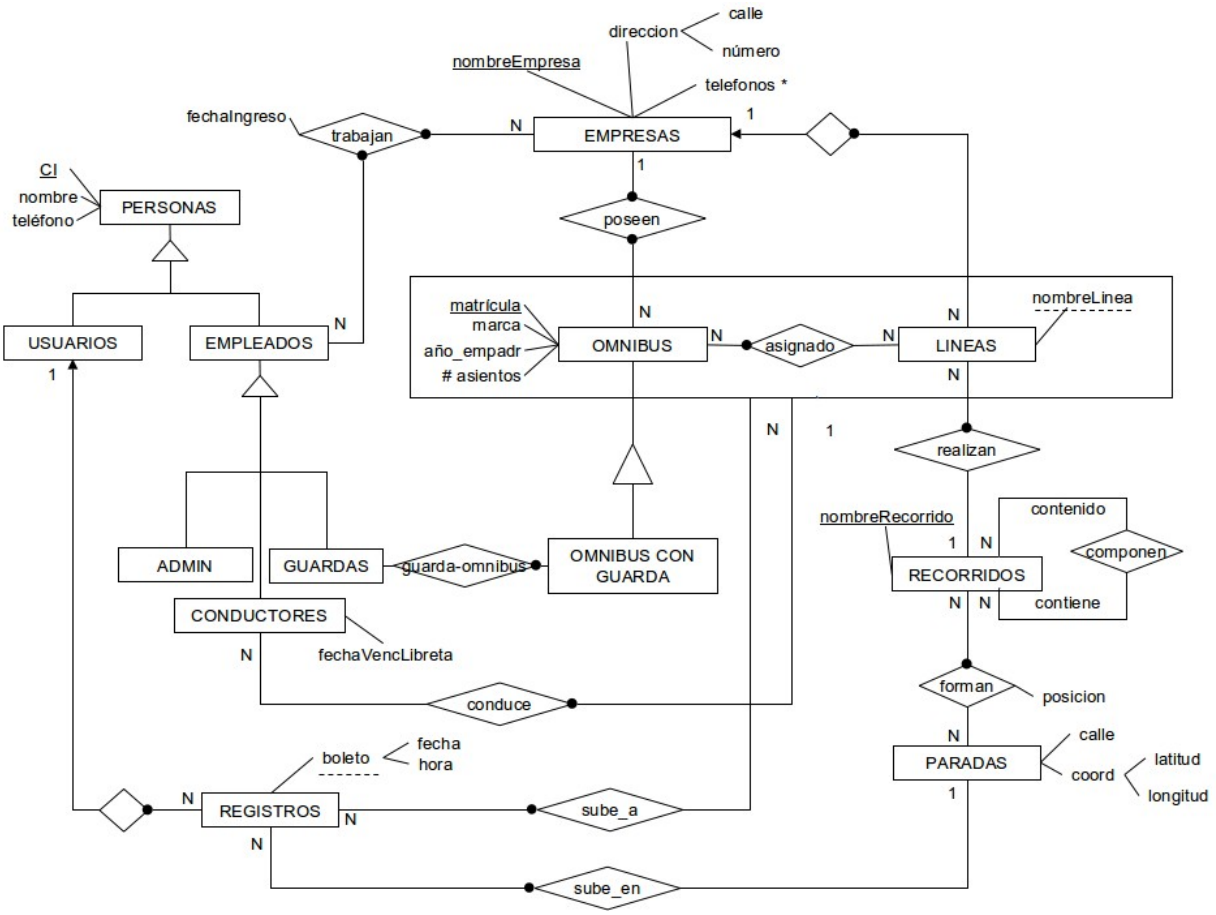
Las empresas organizan los recorridos que ofrecen en líneas. Por ejemplo, la empresa Cutcsa realiza el recorrido Ciudad Vieja – Punta Carretas con la línea 117. El nombre de cada línea está formado por letras y números, e identifica a la línea dentro de la empresa. Cada línea realiza un único recorrido. Las empresas asignan ómnibus de su flota a cada una de las líneas que ofrecen.

En las empresas trabajan empleados, que pueden ser conductores, guardas o personal administrativo. De los empleados se conoce su CI que los identifica, su nombre y teléfono. Cada empleado puede trabajar en más de una empresa, e interesa registrar su fecha de ingreso. Para los conductores se desea saber la fecha en que vence su libreta de conducir. Algunos conductores a veces trabajan de guardas, mientras que el personal administrativo no puede trabajar ni de guarda ni de conductor. Cada ómnibus con asiento para guarda debe tener un guarda asignado. Para minimizar la necesidad de conocer varios recorridos los conductores son asignados a ómnibus de cierta línea.

Algunos de los usuarios del sistema de transporte están registrados. De ellos se conoce su CI que los identifica, su nombre y teléfono. Estos usuarios poseen una tarjeta y por lo tanto es posible registrar en qué fecha y hora se subieron en cierta parada a un ómnibus asociado a una línea en particular. Además interesa registrar si compraron un boleto de una hora o de dos horas.

Se pide Modelo Entidad Relación completo.

SOLUCION (EJ1):



RNE

1. Si un ómnibus es asignado a una línea dicha línea es de la empresa a la que pertenece el ómnibus.
2. Si un conductor conduce un ómnibus de una línea el conductor trabaja para la empresa que posee ese ómnibus
3. Si un usuario sube a un ómnibus de una línea en una parada, esa parada forma el recorrido que realiza la línea.
4. Si un guarda es guarda-omnibus de algún ómnibus con guarda entonces el guarda trabaja para la empresa que posee el ómnibus.
5. Un recorrido no puede estar compuesto por si mismo.
6. Para todo par de paradas p1 y p2 que forman un recorrido la posición es diferente.
7. El año de la fecha de vencimiento de la libreta de chofer tiene que ser mayor o igual al máximo año de ingreso de todas las empresas en las cuales trabaja.
8. ADMIN ∪ CONDUCTORES ∪ GUARDAS = EMPLEADOS
9. ADMIN ∩ CONDUCTORES = Ø
10. ADMIN ∩ GUARDAS = Ø

Ejercicio 2 (25 puntos).

La ciudad universitaria es un conjunto de residencias estudiantiles ubicadas en un mismo predio que dan alojamiento a jóvenes universitarios de todo el país. Cada residencia es construida por una empresa constructora, con fondos de una intendencia departamental. En cada residencia se alojan estudiantes de cualquier departamento del país que estén estudiando en cualquier facultad o escuela de la Universidad de la República.

Se tiene una base de datos que almacena información sobre las residencias y los estudiantes, la cual consta de las siguientes tablas:

RESIDENCIAS (codRes, nomRes, codEmpresa, codIntendencia, añoConst, cantHab)
Esta tabla contiene la información de las residencias de la ciudad universitaria. Para cada residencia se conoce un código que la identifica, su nombre, el código de la empresa que la construyó, el código de la intendencia que financión la obra, el año en que se construyó y la cantidad de habitaciones que tiene.

EMPRESAS (codEmpresa, nomEmpresa, nomTitular, direccion)
Esta tabla contiene información de las empresas que construyen residencias. Para cada una de ellas se conoce un código que las identifica,su nombre, el nombre del titular de la misma y la dirección.

INTENDENCIAS (codIntendencia, nomDepto, nomSección)
Esta tabla contiene datos de las intendencias departamentales. De cada intendencia se conoce un código quela identifica, el nombre del departamento de la intendencia (el cual identifica al departamento) y el nombre de la sección de la intendencia que se ocupa de la construcción de las residencias.

ESTUDIANTES(ciEst, nomEst, deptoOrigen, fchNac)
Esta tabla contiene la información de los estudiantes que se han alojado en alguna residencia de la ciudad universitaria. De cada estudiante se conoce su número de cédula, su nombre, el nombre del departamento de origen y su fecha de nacimiento.

OCUPACION(ciEst, añoCalendario, codRes, becado?, nroHab)
En esta tabla se mantiene información sobre en que residencia se queda cada estudiante en cada año. Además se conoce si esta becado o no ese año en la residencia y que número de habitación ocupa.

En esta base de datos NO hay tablas vacías y se cumplen las siguientes restricciones:

$\Pi_{\text{codEmpresa}}$ (RESIDENCIAS) $\subseteq \Pi_{\text{codEmpresa}}$ (EMPRESAS)

$\Pi_{\text{codIntendencia}}$ (RESIDENCIAS) $\subseteq \Pi_{\text{codEmpresa}}$ (INTENDENCIAS)

Π_{ciEst}(OCUPACION) $\subseteq \Pi_{\text{ciEst}}$(ESTUDIANTES)

Π_{codRes}(OCUPACION) $\subseteq \Pi_{\text{codRes}}$ (RESIDENCIAS)

Resolver la siguiente consulta en Álgebra Relacional:

1. Nombre de los estudiantes que se han alojado en todas las residencias que han sido construidas por empresas cuyo titular es “Pedro Pérez”.

SOL:

Residencias que fueron construidas por las empresas cuyo titular es “Pedro Pérez”.

$$A = \Pi_{codRes}\left(\sigma_{nomTitular = \text{Pedro Pérez}}\left(Empresas * Residencias\right)\right)$$

Nombres de los estudiantes que se han alojado en todas las residencias que están en A.

$$Res = \Pi_{nomEst}\left(\left(\Pi_{ciEst,\,codRes}\left(Ocupacion\right) \,\%\, A\right) * Estudiantes\right)$$

Resolver la siguiente consulta en Cálculo Relacional de tuplas:

2. Nombre de los departamentos tales que todas las residencias financiadas por su intendencia tienen más de 10 habitaciones.
   SOL:

$$Res = \{ <t[nomDepto]> / Intendencias(t) \wedge \forall r (Residencia(r) \wedge \exists i (Intendencias(i) \wedge i[nomDepto] = t[nomDepto] \wedge r[codInst] = i[codInst]) \rightarrow i[nroHab] < 10)) \wedge \exists r (Residencia(r) \wedge \exists i (Intendencias(i) \wedge i[nomDepto] = t[nomDepto] \wedge r[codInst] = i[codInst])) \}$$

Resolver la siguiente consulta en SQL sin usar vistas ni subconsultas en el FROM:

3. Nombre del departamento y el promedio de habitaciones que tienen las residencias que fueron financiadas por la intendencia de ese departamento, considerando solo los departamentos que hayan financiado la construcción de al menos 3 residencias.
   SOL:

```
select nomDepto, avg(nroHab)
from Residencias join Intendencias
where nomDepto=(select nomDepto
                from  Residencias
                group by nomDepto
                having count(codRes) >=3)
group by codRes,nomdepto
```

Escribir en Algebra Relacional una consulta equivalente a la siguiente:

$$\{ t[ciEst] / Ocupacion(t) \wedge becado? = true \wedge \neg (\exists x)(\exists y)(Ocupacion(x) \wedge Ocupacion(y) \wedge x[ciEst] = y[ciEst] \wedge t[ciEst] = x[ciEst] \wedge x[codRes] = y[codRes] \wedge x[añoCalendario] \neq y[añoCalendario]) \}$$

SOL:

Esta consulta devuelve las cédulas de los Estudiantes que fueron becados alguna vez y nunca se alojaron dos años distintos en la misma residencia.

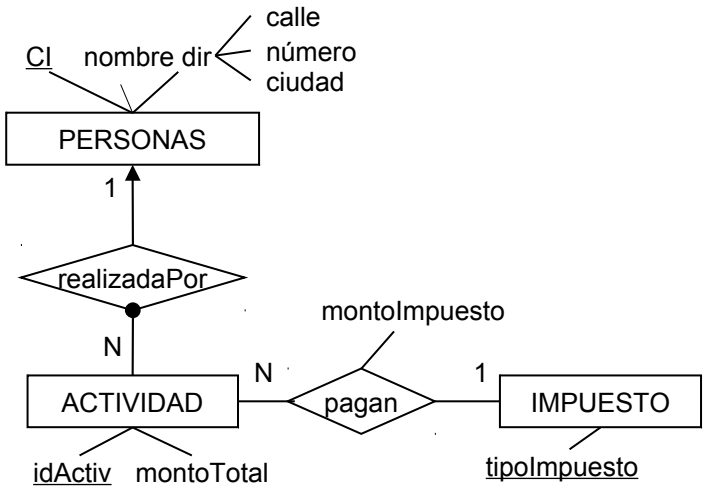
En Algebra sería:

$$A = \Pi_{ciEst}(\sigma_{becado? = true}(Ocupacion))$$

$$Res = A - \Pi_{ciEst}(\rho_{\substack{ciEst = E1, \\ codRes = Res, \\ añoCalendario = año}}(A * Ocupacion) \rhd\lhd_{\substack{ciEst = E1 \\ codRes = Res \\ añoCalendario \neq año}} \Pi_{\substack{ciEst, \\ codRes, \\ añoCalendario}}(A * Ocupacion))$$

Ejercicio 3 (25 puntos).

Considere el siguiente esquema entidad-relación:



a) Determine la Relación Universal que contiene a todos los atributos según el modelo relacional.

b) Determine el conjunto de dependencias funcionales que se deducen del esquema entidad-relación.
c) Construya, siguiendo el algoritmo visto en el curso, una descomposición en 3NF de la relación de la parte (a) según el conjunto de dependencias de la parte (b).
d) A partir del esquema entidad-relación, construya una representación relacional usando las técnicas de pasaje de MER a relacional.
e) Determine en qué forma normal está la representación relacional obtenida en la parte (d).

SOL:

1. **Determine la Relación Universal que contenga a todos los atributos según el modelo relacional.**

   Para esto se ponen todos los atributos en una sola relación desarmando los estructurados.
   R(CI,Nombre,Calle,Nro,Ciudad,IdActiv,MontoTotal,TipoImp,MontoImp)

2. **Determine las dependencias funcionales. Justifique cada una de ellas en función del MER.**

   CI --> Nombre,Calle,Nro,Ciudad (Por la entidad Personas)
   CI,IdActiv-->MontoTotal (Por la entidad débil Actividades)
   CI,IdActiv-->TipoImp,MontoImp (por que la relación pagan es N:1 y MontoImp es atributo de la relación)

3. **Construya una descomposición en tercera forma normal siguiendo el algoritmo visto en el curso.**

   El primer paso es hallar un cubrimiento minimal del conjunto de dependencias.

   El **primer paso** para el cubrimiento, es tener dependencias que sólo tengan un atributo del lado derecho, por lo que, a partir de las dependencias de la parte 2 obtenemos:

   CI --> Nombre
   CI --> Calle
   CI --> Nro
   CI --> Ciudad
   CI,IdActiv-->MontoTotal
   CI,IdActiv-->TipoImp
   CI,IdActiv-->MontoImp

El **segundo paso** es eliminar los atributos redundantes del lado izquierdo. Para eso hay que considerar sólo el segundo grupo de dependencias dado que las otras tienen un único atributo del lado izquierdo. Para esto entonces se deben calcular CI+ y IdACtiv+:

CI+={CI,Nombre,Calle,Nro,Ciudad}
IdActiv+={IdActiv}

Dado que ni CI+ ni IdActiv+ incluyen MontoTotal, TipoImp, MontoImp, ninguno de los dos atributos es redudante en esas dependencias.

El **tercer paso** consiste en eliminar dependencias redudantes: Para esto consideramos las dependencias resultantes del paso 2 (en este caso, las mismas del paso 1) que tengan el mismo atributo del lado derecho. Como no hay dos dependencias que cumplan eso, entonces no hay dependencias redundantes.
Conclusión:

El resultado del primer paso ya es un cubrimiento minimal.

Ahora se debe construir un esquema relacional, dejando una tabla para cada lado izquierdo de las dependencias con todos los atributos del lado derecho. Esto nos da el siguiente esquema relacional:

R1(CI ,Nombre,Calle ,Nro, Ciudad)
R2(CI,IdActiv,MontoTotal,TipoImp,MontoImp)

La clave de la R es CI,IdActiv dado que determina a todos los atributos y por sí solos, los atributos no determinan a todos.

Esa clave está incluída en R2 por lo que tenemos garantía que tenemos JSP.

4. Construya una representación relacional usando el pasaje de mer a relacional.

Primero se construye una tabla por cada entidad, para las entidades débiles se agrega el determinante de la fuerte.

De esta forma nos queda:

Personas(CI,nombre,calle,nro,ciudad)
Actividades(CI,IdActividad,MontoTotal)
Impuestos(TipoImpuesto)

Luego, para cada relación N:N o N:1 no total del lado N se construye una nueva tabla con los atributos de las claves de las tablas de las entidades que participan y los atributos propios de la relación. Cuál es la clave de esa tabla, depende de la cardinalidad de la relación. En este caso:

Pagan(CI,IdActividad,TipoImpuesto,MontoImpuesto)

Observar que parecería que Actividades y Pagan se pueden juntar en una tabla sola dado que tienen la misma clave, pero eso no es así, dado que la relación pagan no es total.

Hay que agregar las depenedencias de inclusión correspondientes:

$$\Pi_{CI}(Actividades) \subseteq \Pi_{CI}(Personas)$$
$$\Pi_{CI,\,IdActividad}(Pagan) \subseteq \Pi_{CI,\,IdActividad}(Actividades)$$
$$\Pi_{TipoImpuesto}(Pagan) \subseteq \Pi_{TipoImpuesto}(Impuestos)$$

De esta forma, el esquema propuesto es:

Personas(CI,nombre,calle,nro,ciudad)
Actividades(CI,IdActividad,MontoTotal)
Impuestos(TipoImpuesto)
Pagan(CI,IdActividad,TipoImpuesto,MontoImpuesto)

Con las dependencias de inclusión anteriores.

5. Determine en qué forma normal está la representación de la parte 4.

Para determinar la forma normal, se deben proyectar el conjunto de dependencias sobre las tablas resultantes (o lo que es lo mismo, asociar a cada tabla las dependencias que surgen de las entidades y relaciones correspondientes)

De esta forma, sobre personas, se cumplen:

CI --> Nombre,Calle ,Nro,Ciudad

Esta dependencia resulta ser de clave por lo que Personas está en 4NF al no haber dependencias multivaluadas.

Sobre actividades, se cumple:

CI,IdActiv-->MontoTotal

Esta dependencia también tiene del lado izquierdo a la clave de la relación por lo que también está en 4NF.

Sobre Pagan se cumple:

CI,IdActiv-->TipoImp,MontoImp

Nuevamente es una dependencia de clave y por tanto está en 4NF.

CONCLUSION: El esquema relacional está en 4NF porque todas sus tablas están en 4NF.

Ejercicio 4 (25 puntos)

Parte a)
Considere las siguientes transacciones:

T1: w1(x) r1(y) w1(z) c1
T2: r2(x) w2(x) r2(y) w2(z) c2

Marque con "si" o "no" en el siguiente cuadro si las historias planteadas son: serializables, recuperables, evitan abortos en cascada (EAC), estrictas.

Id. de Hist.	Historia	Serializ.	Recuper.	EAC	Estricta
H1	r2(x) w1(x) w2(x) r1(y) r2(y) w2(z)w1(z) c1 c2	NO	SI	SI	NO
H2	w1(x) r2(x) w2(x) r2(y)r1(y)w1(z)w2(z) c2 c1	SI	NO	NO	NO

Parte b)
Justifique cada una de las afirmaciones que hizo en la parte anterior.

SOL:
Para H1:
- La historia es Evita Abortos en Cascada, dado que no hay lecturas de una transacción sobre la otra. Es por esto que sin importar cómo confirmen, si la primera aborta, no debería afectar a la segunda. Por esto mismo, la historia es recuperable.
- La historia no es serializable, ya que hay un conflicto de T2 a T1 entre las dos primeras operaciones y un conflicto de T1 a T2 entre la segunda operación y la tercera. Esto hace que el grafo de seriabilidad contenga un ciclo.
- No es estricta, ya que se graba sin confirmación.(Ej: w1(x) se ejecuta antes que el commit de T2)

Para H2:
La historia no es recuperable porque T2 lee de T1 pero confirma antes, por esto no evita abortos en cascada ni es estricta.
Sin embargo, la historia es serializable ya que todos los conflictos tienen primero una operación de T1 y luego una de T2. Observar que sobre Y no hay conflictos porque sólo hay operaciones de lectura.

Parte c)

Reescriba las transacciones T1 y T2 agregándole bloqueos binarios (sólo lock y unlock) de forma que respeten 2PL básico, pero no respeten 2PL conservador ni Estricto.
Justifique.
SOL:
- Para que respete 2PL básico, se deben identificar las fases de crecimiento y decrecimiento de los bloqueos.
- Para que no respete 2PL conservador, no se puede dar que todos los locks estén al ppio de la transacción.
- Para que no respete 2PL conservador, no se puede dar que todos los bloqueos estén al final de la transacción.

De esta forma, la solución es poner las operaciones de bloqueo o desbloqueo donde deben con respecto al item que "protegen", pero no en "las puntas" de la transacción.
Es así que T1 podría ser:
T1: l1(x) w1(x) l1(y) l1(z) r1(y) u1(x) u1(y) w1(z) u1(z)c1
T2:l2(x) r2(x) w2(x)l2(y) l2(z) r2(y)u2(x)u2(y) w2(z)u2(z) c2

Parte d)
Indique cuál era el objetivo y qué elemento además de los bloqueos se necesita considerar para implementar los algoritmos de wound-wait.
SOL:
El objetivo del algoritmo de wound-wait es la prevención de deadlock y el elemento extra que necesita considerar es un timestamp de lectura y otro de escritura para cada item y un timestamp para cada transacción.