Algunas soluciones a los ejercicios de normalización

axiomas de Armstrong

Reflexiva : $\forall\ \mathbf{Y} \subseteq \mathbf{X} \Rightarrow \mathbf{X} \rightarrow \mathbf{Y}$ *[R1]*

Aumento : $\mathbf{X} \rightarrow \mathbf{Y} \Rightarrow \mathbf{ZX} \rightarrow \mathbf{ZY}$ *[R2]*

Transitiva : $\mathbf{X} \rightarrow \mathbf{Y}$ *y* $\mathbf{Y} \rightarrow \mathbf{Z} \Rightarrow \mathbf{X} \rightarrow \mathbf{Z}$ *[R3]*

reglas derivadas

union : $\mathbf{X} \rightarrow \mathbf{Y}$ *y* $\mathbf{X} \rightarrow \mathbf{Z} \Rightarrow \mathbf{X} \rightarrow \mathbf{YZ}$ *[R4]*

descomposición : $\mathbf{X} \rightarrow \mathbf{Y}$ *y* $\mathbf{Z} \subseteq \mathbf{Y} \Rightarrow \mathbf{X} \rightarrow \mathbf{Z}$ *[R5]*

pseudotransitividad : $\mathbf{X} \rightarrow \mathbf{Y}$ *y* $\mathbf{WY} \rightarrow \mathbf{Z} \Rightarrow \mathbf{WX} \rightarrow \mathbf{Z}$ *[R6]*

---

1) Indicar la forma normal de la siguiente relación y normalizarla si fuera necesario:

ParcelaRustica(refCatastral, municipio, parcela, codMunicipio)
con las siguientes dependencias funcionales:
DF1: *{refCatastral}* → *{municipio}*
DF2: *{refCatastral}* → *{parcela}*
DF3: *{municipio}* → *{codMunicipio}*
DF4: *{codMunicipio}* → *{municipio}*
DF5: *{municipio, parcela}* → *{refCatastral}*

Se supone que toda relación está en 1FN (restricción inherente al modelo relacional) y que el conjunto de atributos siempre constituye una clave de la relación (no puede haber tuplas repetidas).

Aplicando *[R3]* a DF1 y DF3 se obtiene la DF6: *{refCatastral}* → *{codMunicipio}*. Con *[R4]* sobre DF1, DF2, DF6, y *[R1]*, se obtiene la clave candidata K1 = (refCatastral). Aplicando *[R3]* a este resultado con DF5, se obtiene la clave candidata K3 = (municipio, parcela). Finalmente, si se aplica *[R2]* a DF4 se puede obtener *{codMunicipio, parcela}* → *{municipio, parcela}* que con DF5 y *[R3]* permite obtener la última clave candidata: K2 = (codMunicipio, parcela)

Como ∄ ningún atributo no-primo, se puede concluir que está en 3FN

test de 3FNBC

De las claves candidatas no se deducen DF3 y DF4 por lo que no está en 3FNBC o, dicho de otra manera, *{municipio}* → *{codMunicipio}* pero *municipio* no es, ni contiene, una clave candidata. Lo mismo sucede con *{codMunicipio}* → *{municipio}*

Para normalizar **ParcelaRustica** se puede realizar la siguiente descomposición:

datParcelaR(refCatastral, municipio, parcela) con DF1, DF2 y DF5 ⇒ K1 y K3

datMunic(codMunicipio, municipio) con DF3 y DF4 ⇒ K4 = (codMunicipio) y K5 = (municipio)

Se puede comprobar que la descomposición es sin pérdidas y se preservan las dependencias funcionales.

test de 4FN

Como no hay ninguna dependencia multivaluada no trivial en ninguna relación, la BD está en 4FN

test de 5FN

Como no hay ninguna descomposición no aditiva y sin pérdidas de las relaciones, la BD está en 5FN

Así pues, la BD normalizada resultante es:

datParcelaR(refCatastral, municipio, parcela)

datMunic(codMunicipio, municipio)

4) Indicar la forma normal de la siguiente relación y normalizarla si fuera necesario:

viaje(idRuta, horaSalida, fecha, matricula, DNIchofer, ApellNombre)
con las siguientes dependencias funcionales:
DF1: *{idRuta, horasalida}* → *{matricula, DNIchofer}* DF2: *{DNIchofer}* → *{ApellNombre}*

Se supone que toda relación está en 1FN (restricción inherente al modelo relacional) y el conjunto de atributos siempre constituye una clave de la relación.

Aplicando *[R3]* a DF1 y DF2 se obtiene la DF3: *{idRuta, horasalida}* → *{matricula, DNIchofer, ApellNombre}*. Con *[R2]* sobre DF1 y DF3, se obtiene la única clave candidata K1 = (idRuta, horaSalida, fecha)

Los atributos no-primos (∉ a ninguna clave candidata) son: matricula,DNIchofer,ApellNombre

test de 2FN

La DF3 especifica atributos no-primos que dependen funcionalmente de parte de la clave candidata, K1, por lo que la relación no está en 2FN

Para transformar la BD a otra en la que todas sus relaciones estén en 2FN se puede hacer la siguiente descomposición no aditiva y sin pérdidas:

viajeReal(idRuta, horaSalida, fecha)

viajeD(idRuta, horaSalida, matricula, DNIchofer, ApellNombre) con DF3

Resulta simple comprobar que ambas relaciones están en 2FN y se preservan todas las DF.

test de 3FN

En **viajeD** la DF2 especifica que el atributo no-primo *{ApellNombre}* depende funcionalmente de otro atributo no-primo *{DNIchofer}*, por lo que no verifica la 3FN

Para transformar **viajeD** en relaciones que verifiquen la 3FN se puede hacer la siguiente descomposición no aditiva y sin pérdidas (es simple comprobar que las nuevas relaciones están en 3FN):

viajeDiario(idRuta,horaSalida,matricula,DNIchofer) con DF1⇒ K2=(idRuta,horaSalida)

conductor(DNIchofer, ApellNombre) con DF2 ⇒ K3=(DNIchofer)

test de 3FNBC

Puesto que todas las DF son consecuencia de las claves candidatas K2 y K3, respectivamente, todas las relaciones obtenidas están en 3FNBC

test de 4FN

Como no hay ninguna dependencia multivaluada no trivial en ninguna relación, la BD está en 4FN

test de 5FN

Como no hay ninguna descomposición no aditiva y sin pérdidas, de las relaciones, la BD está en 5FN

Así pues, la BD normalizada resultante es:

viajeReal(idRuta, horaSalida, fecha)
viajeDiario(idRuta, horaSalida, matricula,DNIchofer)
conductor(DNIchofer, ApellNombre)

Nota: Obsérvese que se ha obtenido la misma solución que se propuso en el ejercicio del TURYBUS

7) Indicar la forma normal de la siguiente relación y normalizarla si fuera necesario:

prestamo(numSocio, nombSocio, ISBN, fechaPrestamo, editorial, país)
con las siguientes dependencias funcionales:
DF1: *{ISBN } → {editorial}*
DF2: *{numSocio, ISBN} → {fechaPrestamo}*
DF3: *{editorial} → {país}*
DF4: *{numSocio} → {nombSocio}*
DF5: *{nombSocio} → {numSocio}*

Se supone que toda relación está en 1FN (restricción inherente al modelo relacional) y el conjunto de atributos siempre constituye una clave de la relación.

Aplicando las reglas de DF *[R1]* sobre DF2 y después *[R3]* con DF1, DF3 y DF4 resulta simple obtener las claves candidatas K1 = (numSocio, ISBN) y K2 = (nombSocio, ISBN)

Los atributos no-primos (∉ a ninguna clave candidata) son: fechaPrestamo, editorial, país

test de 2FN

aplicando *[R3]* a DF1, DF3 se obtiene DF6: *{ISBN } → {país}* por lo que DF1 y DF6 no verifican la 2FN (son atributos no-primos que dependen funcionalmente de parte de una clave candidata, K1, y también de K2)

Para transformar la BD a otra en la que todas sus relaciones estén en 2FN se puede hacer la siguiente descomposición no aditiva y sin pérdidas:

prestamo1(numSocio, nombSocio, ISBN, fechaPrestamo) con DF2, DF4 y DF5 ⇒ K1 y K2

libro1(ISBN, editorial, país) con DF1 y DF3 ⇒ clave candidata, K3 = (ISBN)

Resulta simple comprobar que ambas relaciones están en 2FN y se preservan todas las DF.

test de 3FN

En **prestamo1** el único atributo no-primo es fechaPrestamo por lo que no puede haber dependencias transitivas (∄ ningún otro atributo no-primo)

En **libro1** los únicos atributos no-primos son editorial y país. DF3 representa una dependencia transitiva sobre esos atributos, por lo que no verifica la 3FN

Para transformar **libro1** en relaciones que verifiquen la 3FN se puede hacer la siguiente descomposición no aditiva y sin pérdidas (es simple comprobar que las nuevas relaciones están en 3FN):

libro(ISBN, editorial) con DF1 ⇒ clave candidata K3 = (ISBN)

editor(editorial, país) con DF3 ⇒ clave candidata K4 = (editorial)

test de 3FNBC

En **prestamo1** las claves candidatas son K1 y K2, pero de ellas no se deducen DF4 y DF5 por lo que no está en 3FNBC o, dicho de otra manera, *{numSocio} → {nombSocio}* pero *numSocio* no es, ni contiene, una clave candidata. Lo mismo sucede con *{nombSocio} → {numSocio}*

Para normalizar **prestamo1** se puede realizar la siguiente descomposición:

prestar (numSocio, ISBN, fechaPrestamo) con DF2 ⇒ K1

socio(numSocio, nombSocio) con DF4 y DF5 ⇒ claves K4 = (numSocio) y K5 = (nombSocio)

Se puede comprobar que la descomposición es sin pérdidas y se preservan las dependencias funcionales.

test de 4FN

Como no hay ninguna dependencia multivaluada no trivial en ninguna relación, la BD está en 4FN

test de 5FN

Como no hay ninguna descomposición no aditiva y sin pérdidas, de las relaciones, la BD está en 5FN

Así pues, la BD normalizada resultante es:

prestar (numSocio, ISBN, fechaPrestamo);
libro(ISBN, editorial)
socio(numSocio, nombSocio)
editor(editorial, país)

8) Indicar la forma normal de la siguiente relación y normalizarla si fuera necesario:

gestorPersonal(idCliente, idSucursal, idEmpleado)
con las siguientes dependencias funcionales:
DF1: *{idCliente, idSucursal} → {idEmpleado}* DF2: *{idEmpleado} → {idSucursal}*

Se supone que toda relación está en 1FN (restricción inherente al modelo relacional) y el conjunto de atributos siempre constituye una clave de la relación.

con *[R1]* y *[R2]* es fácil obtener las claves candidatas: K1=(idCliente,idSucursal) y K2=(idCliente,idEmpleado)

Como ∄ ningún atributo no-primo, se puede concluir que está en 3FN

test de 3FNBC

De las claves candidatas no se deduce la DF2 por lo que no está en 3FNBC o, dicho de otra manera, *{idEmpleado} → {idSucursal}* pero *idEmpleado* no es, ni contiene, una clave candidata.

Para intentar normalizar la relación, se pueden realizar las siguientes descomposiciones:

asignEmpl(idCliente, idEmpleado) y **empleado**(idEmpleado, idSucursal)

que no preserva la DF1 y, por tanto, no se considera adecuada

asignSuc(idCliente, idSucursal) y **asignEmpl**(idCliente, idEmpleado)

que no preserva la DF2, por lo que tampoco se considera adecuada

En resumen, como no se encuentra ninguna descomposición que preserve todas las DF, podría plantearse como solución el añadir a la relación original una nueva relación: **empleado**(idEmpleado, idSucursal) que se utilizaría para implementar la restricción que especifica la DF2. Es decir,

gestorPersonal(idCliente, idSucursal, idEmpleado)
empleado(idEmpleado, idSucursal)

y la restricción (disparador) de que el empleado asignado al cliente en **gestorPersonal** debe pertenecer a la sucursal.

idCliente	idSucursal	idEmpleado
Pepe	Coso	empl_1
Pepe	Universidad	empl_2
Juan	Coso	empl_3
Antonio	Coso	empl_3
María	Universidad	empl_2

fig.1 Ejempo de instancia de la BD

ejemplo: supóngase la instancia de la BD de la fig.1

si se efectúa la primera descomposición propuesta:

idCliente	idEmpleado
Pepe	empl_1
Pepe	empl_2
Juan	empl_3
Antonio	empl_3
María	empl_2

y

idEmpleado	idSucursal
empl_1	Coso
empl_2	Universidad
empl_3	Coso

¿ qué sucede al añadir [Pepe, empl_3] ?

si se efectúa la segunda descomposición propuesta, tendríamos:

idCliente	idSucursal
Pepe	Coso
Pepe	Universidad
Juan	Coso
Antonio	Coso
María	Universidad

y

idCliente	idEmpleado
Pepe	empl_1
Pepe	empl_2
Juan	empl_3
Antonio	empl_3
María	empl_2

¿ qué sucede al añadir [Pepe, empl_3] ?

Ejercicio: Especifique en detalle las soluciones que utilizaría para garantizar la integridad de la BD en las diferentes soluciones propuestas.

2) **matricular**(NIP, fechaNac, asignatura)
con la siguiente dependencia funcional: DF1: *{NIP} → {fechaNac}*

3) **matricular**(NIP, ApNombre, asignatura)
con las siguientes dependencias funcionales:
DF1: *{ApNombre} → {NIP}* DF2: *{NIP} → {ApNombre}*

5) **billete**(idViaje, idRuta, horaSalida, fecha, DNIpasajero)
con las siguientes dependencias funcionales:
DF1: *{idRuta, horasalida, fecha} → {idViaje}* DF2: *{idViaje} → {idRuta, horasalida, fecha}*

6) **suministrar**(clvProv, nombProv, clvPieza, nombPieza, cantidad)
con las siguientes dependencias funcionales: DF1: *{clvProv, clvPieza} → {cantidad}*
DF2: *{clvProv } → {nombProv}* DF3: *{clvPieza} → {nombPieza}*
DF4: *{nombProv} → {clvProv}* DF5: *{nombPieza} → {clvPieza}*