**Ejercicios Tema 3. NORMALIZACIÓN BASES DE DATOS**

1. **Define Base de Datos Relacional.**

Una base de datos relacional es una base de datos que está bien diseñada, es decir, una base de datos percibida por el usuario como una colección de relaciones normalizadas de diversos grados que varían con el tiempo. Estas bases operan conceptualmente sobre tablas completas mediante la especificación de operaciones sobre ellas. Para realizarlas se debe hacer el modelo E-R y pasarlo a modelo relacional, tras lo cual debemos asegurarnos que el diseño esté normalizado.

1. **¿Qué es la Normalización?**

La normalización es una técnica para agrupar la información de los datos en diferentes conjuntos de modo que se faciliten los procesos de manipulación, asegurando en todo momento un diseño que minimice redundancias, pérdidas de información y problemas de representación. Para que un diseño esté normalizado hay que verificar que cada tabla del mismo se encuentra en 3º forma normal, por lo que también debe de estar en 2º y 1º.

1. **Define la Primera Forma Normal.**

Es aquella que se cumple cuando cada atributo de una tabla contiene un valor único. Es decir, los dominios no tienen elementos que sean conjuntos y todos los valores son atómicos.

1. **Define Segunda Forma Normal.**

Es aquella que se cumple cuando una tabla está en 1FN y además todos los atributos no primarios de esta dependen funcionalmente de manera plena de la clave primaria. Esto implica que los atributos no deben depender solo de una parte de una clave primaria compuesta.

1. **Define Tercera Forma Normal.**

Es aquella que se cumple cuando una tabla esta en 2FN, y por ende también en primera, y además no existen atributos no primarios que dependan transitivamente de la clave primaria. Es decir, ningún atributo que no sea clave puede depender de otro atributo no clave.

1. **Explica cuando una tabla NO está en Primera Forma Normal, los inconvenientes que puede tener y como arreglarías el modelado para que estuviera en Primera Formal Normal. Pon un ejemplo.**

Una tabla no está en 1FN cuando un atributo contiene valores multivaluados o compuestos, como por ejemplo en una tabla de cliente tenemos varios teléfonos dentro de un mismo atributo. Los inconvenientes que presenta son dificultad en consultas y en manipulación de datos. Las posibles soluciones, usando el ejemplo anterior, supondría crear varios atributos de teléfono en caso de que conozcamos el número exacto de teléfonos o solo queramos conocer algunos, para ello se usaría NULL y NOT NULL. La otra opción, en caso de no conocer o no buscar un número exacto de teléfonos supondrán establecer el atributo como multivaluado, es decir, crear una tabla independiente para teléfonos que se relacione con la de clientes.

1. **Explica cuando una tabla NO está en Segunda Forma Normal, los inconvenientes que puede tener y como arreglarías el modelado para que estuviera en Segunda Formal Normal. Pon un ejemplo.**

Una tabla no está en 2FN cuando existen dependencias parciales en una clave compuesta, es decir cuando hay atributos que dependen solo de una parte de la clave primaria compuesta. Como inconvenientes da lugar a redundancia y problemas de actualización posibles. Un ejemplo sería tener dentro de una tabla pedido que contiene los campos NumPedido y CodProducto, siendo estos PK compuesta, además de NomProducto y cantidad. En este caso no está en 2FN ya que NomProducto depende de una parte de la PK compuesta únicamente, de CodProducto, para arreglarlo se debe crear una nueva tabla, producto, la cual almacene este valor y envía la PK como FK a la tabla anterior, sin tener ahora el valor de NomProducto en ella.

1. **Explica cuando una tabla NO está en Tercera Forma Normal, los inconvenientes que puede tener y como arreglarías ese modelado para que estuviera en Tercera Formal Normal. Pon un ejemplo.**

Una tabla no esta en 3FN si contiene atributos que no sean clave primaria que dependan transitivamente de la clave primaria. Esto ocurre cuando un atributo no clave depende de otro no clave, en lugar de depender directamente de la clave primaria. Todo esto genera mucha redundancia y problemas de actualización, ya que un cambio en el atributo intermedio requiere de multiples actualizaciones. Un ejemplo sería tener una tabla vendedor que contuviese los campos CodVendedor, ciudad y pais, en ella pais dependería transitivamente de CodVendedor a través de ciudad, para arreglarlo se debe de crear una tabla nueva, ciudad, que almacene estos valores. También en este caso podría crearse una tabla adicional, para pais que enviase FK a la de ciudad, en vez de almacene ambos valores en la misma tabla.

1. **¿En qué forma normal no se encuentra la siguiente relación? Aplica la regla correspondiente y arregla la relación dividiéndola en las relaciones necesarias.**

**R: (NumMat, NomAlu, ApeAlu, DirAlu, CodCiu, NomCiu)**

La relación no está en 3FN porque NomCiu depende transitivamente de CodCiu en lugar de depender directamente de la clave primaria NumMat. Para solucionarlo la tabla se divide en 2, una que almacene la información del alumno: NumMat, NomAlu, ApeAlu, DirAlu y CodCiu. La otra tabla alamcenerá la información de las ciudades, conteniendo CodCiu y NomCiu, enviando como FK a la tabla de alumnos CodCiu.

1. **¿En qué forma normal no se encuentra la siguiente relación? Explica que debe cumplir una relación para estar en dicha forma normal. Aplica la regla correspondiente y arregla la relación dividiéndola en las relaciones necesarias.**

**R: (NumMat,CodAsignatura,NomAsignatura,NomAlu,ApeAlu,DNIAlu, NotaAluAsig)**

La relación no está en 2FN ya que el atributo NomAsignatura depende solo de una parte de la PK compuesta, de CodAsignatura. Para solucionarlo se hacen 3 tablas:

* alumno (almacena la información de los alumnos): NumMat, NomAlu, ApeAlu y DNIAlu
* asignatura (almacena la información de las asignaturas): CodAsignatura y NomAsignatura.
* nota (almacena la nota de cada alumno en cada asignatura): NumMat, CodAsignatura y NotaAluAsig.