TC2005B Construcción de software y toma de decisiones

Normalización

Introducción

Uno de los retos en el diseño de la base de datos es el de obtener una estructura estable y lógica tal que el sistema de base de datos no sufra de **anomalías** de almacenamiento y el modelo lógico pueda **modificarse fácilmente** para admitir nuevos requerimientos.

Una base de datos implantada sobre un modelo **bien diseñado** tiene mayor esperanza de vida aun en un ambiente **dinámico**, que una base de datos con un diseño pobre. En promedio, una base de datos experimenta una reorganización general cada **seis** años, dependiendo de lo dinámico de los requerimientos de los usuarios. Una base de datos bien diseñada tendrá un buen desempeño aunque aumente su tamaño, y será lo suficientemente flexible para incorporar nuevos requerimientos o características adicionales.

Existen diversos riesgos en el diseño de las bases de datos relacionales que afecten la funcionalidad de la misma, los riesgos generalmente son la **redundancia de información y la inconsistencia de datos**.

La normalización es el **proceso de simplificar la relación entre los campos de un registro**. Por medio de la normalización un conjunto de datos en un registro se reemplaza por varios registros que son más simples y predecibles y, por lo tanto, más manejables.

Razones para llevar a cabo una normalización

- Estructurar los datos de forma que se puedan representar las **relaciones pertinentes** entre los datos.
- Permitir la **recuperación sencilla** de los datos en respuesta a las solicitudes de consultas y reportes.
- **Simplificar el mantenimiento** de los datos actualizándolos, insertándolos y borrándolos.

Cuando trabajo con BD

① En el proyecto se crea la BD desde cero

Recordemos:

planear → 1) Identificar necesidad
2) Identificar requisitos de información

analizar → 3) Elaborar diagrama de contexto (entrevista)
4) MER / EMER

diseño → 5) MR (Traducir del MER)

② Cuando la "BD" ya existe

⇒ normalizarlo

La siguiente relación es utilizada por el departamento de cobros de una empresa que ofrece el servicio de telecable, En ella se tiene una representación de los pagos de servicios mensuales de sus contratantes:

Servicios(cliente, domicilio y estado , año, rentabasica 1, servicios adicionales 1, rentabasica 2, servicios adicionales 2, rentabasica 3, servicios adicionales 3, ... rentabasica 12, servicios adicionales 12)

1NF

Servicios(cliente, domicilio, estado , año, mes, rentabasica, servicios adicionales)

2NF

Servicios(cliente, domicilio, estado , año, mes, rentabasica, servicios adicionales)

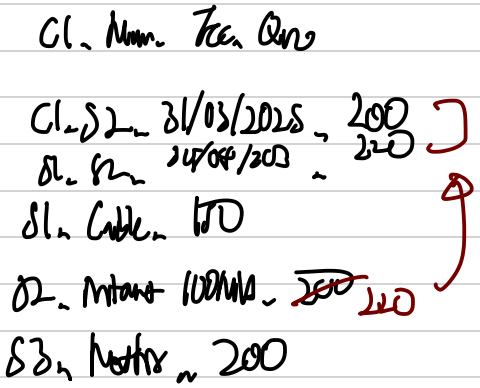
PK -> Servicios(cliente, domicilio , año, mes)

3NF

Servicios(noContrato, nombre, domicilio, estado)

Renta(noContrato, idServicio, fecha, Monto)

Servicios(idServicio, descripción, tarifa)



La siguiente relación es utilizada por el departamento de cobros de una empresa que ofrece el servicio de telecable, En ella se tiene una representación de los pagos de servicios mensuales de sus contratantes:

Servicios(cliente, domicilio y estado , año, rentabasica 1, servicios adicionales 1, rentabasica 2, servicios adicionales 2, rentabasica 3, servicios adicionales 3, ... rentabasica 12, servicios adicionales 12)

1NF

Servicios(cliente, domicilio, estado , año, mes, rentabasica, servicios adicionales)

2NF

Servicios(cliente, domicilio, estado , año, mes, rentabasica, servicios adicionales)

PK -> Servicios(cliente, domicilio , año, mes)

3NF

Servicios(noContrato, nombre, domicilio, estado)

Renta(noContrato, idServicio, fecha, Monto)

Servicios(idServicio, descripción, tarifa)

- **Reducir la necesidad de reestructurar** o reorganizar los datos cuando surjan nuevas aplicaciones.

Pasos para hacer una normalización

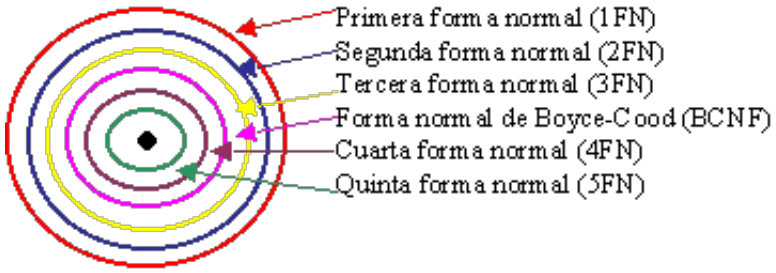
1. Descomponer todos los grupos de datos en registros bidimensionales.
2. Eliminar todas las relaciones en la que los datos no dependan completamente de la llave primaria del registro.
3. Eliminar todas las relaciones que contengan dependencias transitivas.

La teoría de normalización tiene como fundamento el concepto de **formas normales**; se dice que una relación está en una determinada forma normal si satisface un conjunto de restricciones.

Formas normales

Son las técnicas para prevenir las **anomalías** en las tablas. Dependiendo de su estructura, una tabla puede estar en primera forma normal, segunda forma normal o en cualquier otra.

Relación entre las formas normales:



Primera forma normal

Una relación R(Tabla) se encuentra en 1FN si y solo sí por cada renglón columna contiene valores atómicos.

Abreviada como **1FN**, se considera que una relación se encuentra en la primera forma normal cuando cumple lo siguiente:

- Las celdas de las tablas poseen **valores simples** y no se permiten grupos ni arreglos repetidos como valores, es decir, contienen un solo valor por cada celda.
- Todos los ingresos en cualquier columna (atributo) deben ser del **mismo tipo**.
- Cada columna debe tener un **nombre único**, el orden de las columnas en la tabla no es importante.
- Dos filas o renglones de una misma tabla **no deben ser idénticas**, aunque el orden de las filas no es importante.

Por lo general la mayoría de las relaciones cumplen con estas características, así que podemos decir que la mayoría de las relaciones se encuentran en la primera forma normal.

Para ejemplificar como se representan gráficamente las relaciones en primera forma normal consideremos la relación alumno cursa materia cuyo diagrama E-R es el siguiente:

Como esta relación maneja valores atómicos, es decir un sólo valor por cada uno de los campos que conforman a los atributos de las entidades, ya se encuentra en **primera forma normal**, gráficamente así representamos a las relaciones en **1FN**.

Otra notación utilizada frecuentemente es la siguiente:

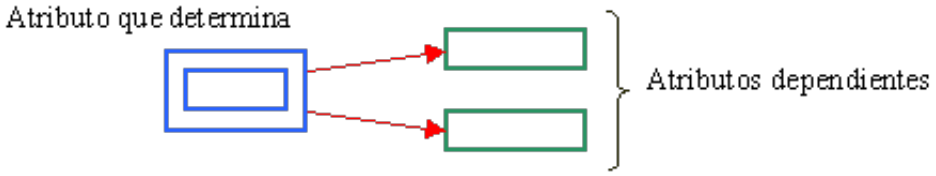
Alumno (Control, Nombre, Esp)

Materia (Clave, NomM, Creditos)

Como podemos ver cada columna guardará sólo un valor del mismo tipo de dato, por lo que ambas relaciones cumplen con las restricciones de la **1FN**.

Segunda forma normal

Para definir formalmente la segunda forma normal requerimos saber qué es una dependencia funcional: Consiste en edificar que atributos dependen de otro(s) atributo(s).

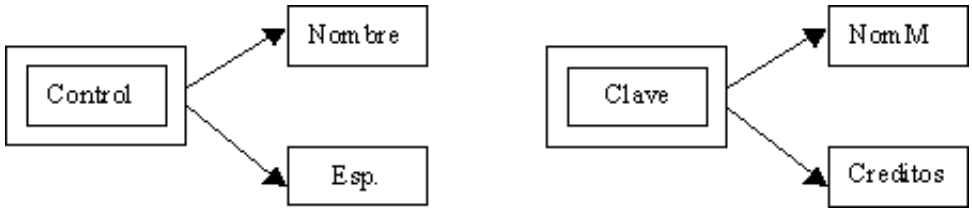


Una relación R está en 2FN si y sólo si está en 1FN y los atributos no primos (llaves) dependen funcionalmente de la llave primaria.

Una relación se encuentra en segunda forma normal, cuando cumple con las reglas de la primera forma normal y todos sus atributos que no son claves (**llaves**) dependen por completo de la clave

De acuerdo con está definición, cada tabla que tiene un atributo único como clave, está en segunda forma normal.

La segunda forma normal se representa por dependencias funcionales como:



Nota: las **llaves primarias** están representadas con **doble cuadro**, las flechas nos indican que de estos atributos se puede referenciar a los otros atributos que **dependen funcionalmente** de la **llave primaria**.

Tercera forma normal

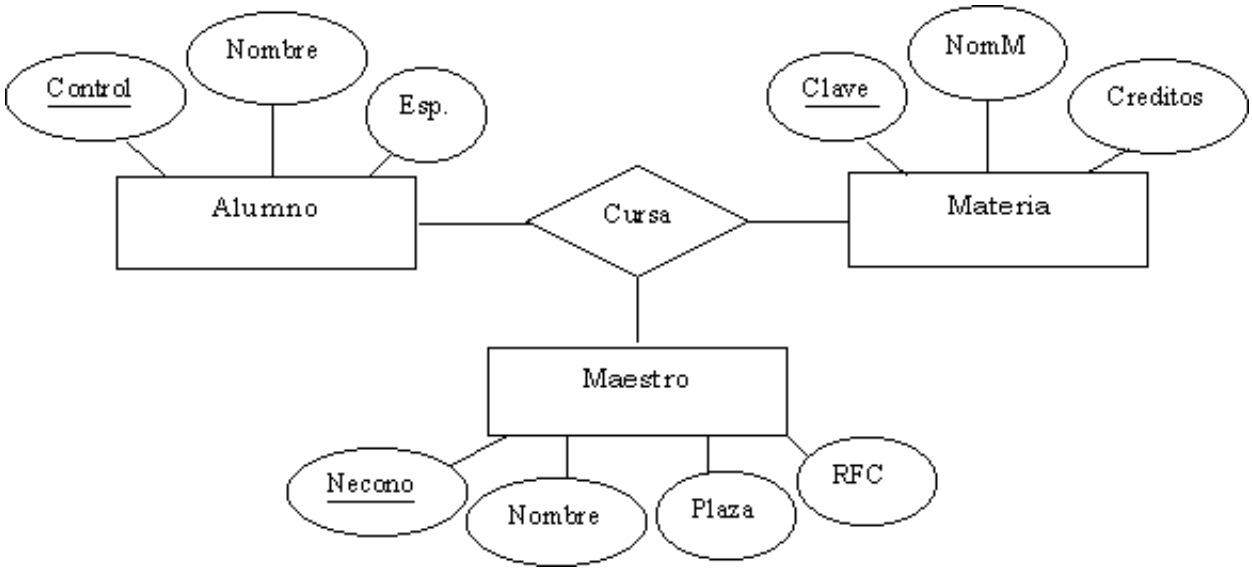
Para definir formalmente la **3FN** necesitamos definir **dependencia transitiva**:

*En una afinidad **(tabla bidimensional)** que tiene por lo menos **3 atributos** (A,B,C) en donde A determina a B, B determina a C pero **no** determina a A.*

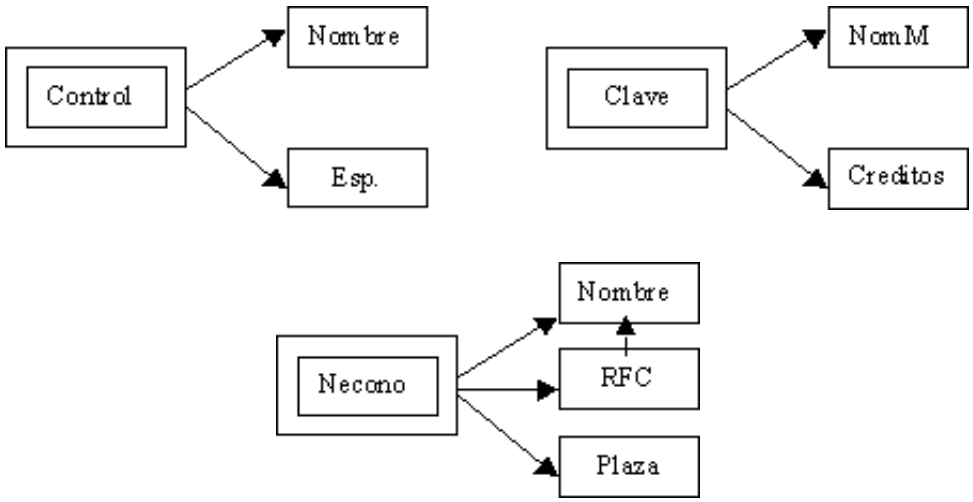
Una relación R está en 3FN si y sólo si esta en 2FN y todos sus atributos no primos (llaves) dependen no transitivamente de la llave primaria.

Consiste en eliminar **la dependencia transitiva** que queda en una segunda forma normal, en pocas palabras una relación esta en tercera forma normal **si está en segunda forma normal y no existen dependencias transitivas entre los atributos,** nos referimos a dependencias transitivas cuando

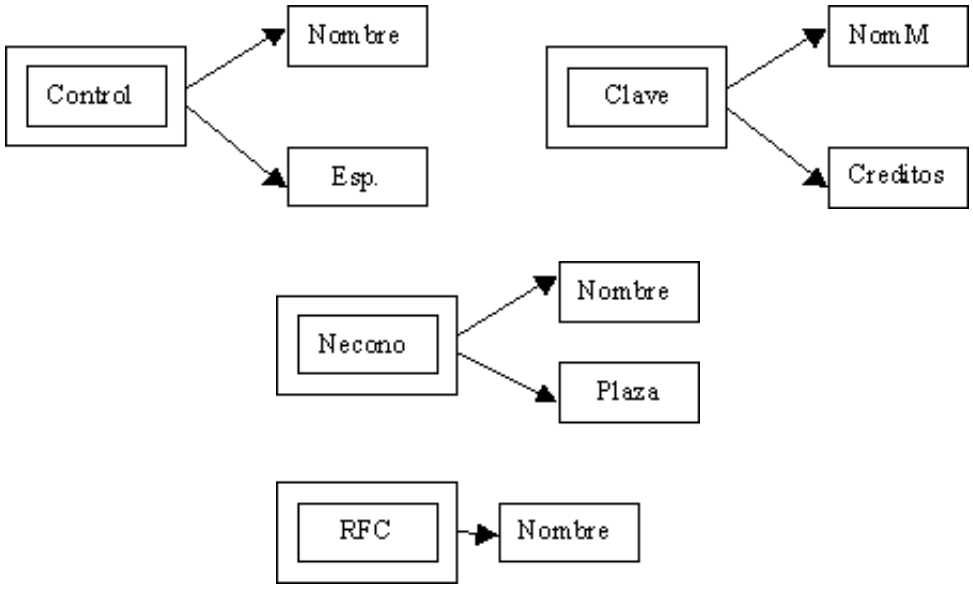
existe más de una forma de llegar a referencias a un atributo de una relación.



Tenemos la relación **alumno-cursa-materia**manejada anteriormente, pero ahora consideramos al elemento **maestro**, gráficamente lo podemos representar de la siguiente manera:



Podemos darnos cuenta que se encuentra graficado en **segunda forma normal**, es decir que todos los atributos llave están indicados en doble cuadro indicando los atributos que dependen de dichas llaves, sin embargo en la llave **Necono** (numero economico o numero de nómina) tiene como dependientes a **3 atributos** en el cual el **nombre** puede ser referenciado por dos atributos: **Necono y RFC** (Existe dependencia transitiva), podemos solucionar esto aplicando la tercera forma normal que consiste en **eliminar** estas dependencias separando los atributos, entonces tenemos:



Forma normal de Boyce Codd

Determinante: Uno o más atributos que, de manera funcional, determinan otro atributo o atributos.

En la dependencia funcional (A,B)-->C, **(A,B) son los determinantes**.

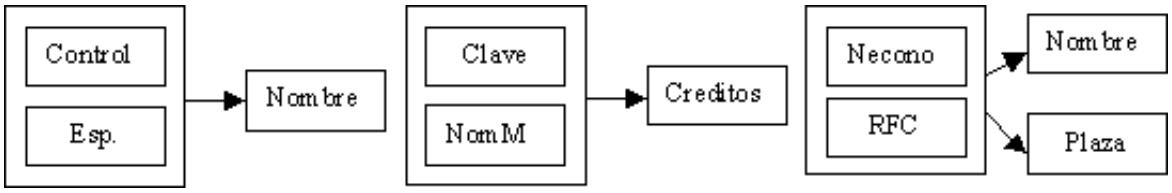
Una relación R esta en FNBC si y sólo si cada determinante es una llave candidata.

Recordando: Algunas relaciones tienen una sola **llave posible**, mientras que otras presentan varias **llaves**. Al conjunto de llaves de una relación se le conoce como conjunto de **llaves candidatas**. De entre las **llaves candidatas**, debe seleccionarse una de ellas para definir la **llave primaria** de la relación; una vez seleccionada la **llave primaria**, el resto de las **llaves candidatas** se denominan **llaves alternas**

Denominada por sus siglas en ingles como **BCNF**; Una tabla se considera en esta forma **si y sólo sí cada determinante o atributo es una llave candidata**.

Continuando con el ejemplo anterior, si consideramos que en la entidad **alumno** sus atributos **control y Esp.** nos puede hacer referencia al atributos **nombre**, entonces decimos que dichos atributos pueden ser **llave candidata**.

Gráficamente podemos representar la forma normal de **Boyce Codd** de la siguiente forma:



Obsérvese que a diferencia de la tercera forma normal, agrupamos **todas las llaves candidatas para formar una global** (representadas en el recuadro) las cuales hacen referencia a los atributo que no son **llaves candidatas**.

Cuarta forma normal

Un esquema de relaciones R está en 4FN con respecto a un conjunto D de dependencias funcionales y de valores múltiples sí, para todas las dependencias de valores múltiples en D de la forma X->->;Y, donde X<=R y Y<=R, se cumple por lo menos una de estas condiciones:

- **X->->Y es una dependencia de valores múltiples trivial.**
- **X es una superllave del esquema R.**

Recordando:
Cualquier conjunto de columnas que contiene un **subconjunto de columnas que es llave**, también es **llave;** a este tipo de conjuntos de columnas que contienen una **llave** se les denomina **superllaves**.

Para entender mejor aún esto consideremos una afinidad (tabla) llamada **estudiante** que contiene los siguientes atributos: **Clave, Especialidad, Curso** tal y como se demuestra en la siguiente figura:

Clave	Especialidad	Curso
S01	Sistemas	Natación
S01	Bioquimica	Danza
S01	Sistemas	Natación
B01	Bioquimica	Guitarra
C03	Civil	Natación

Suponemos que los **estudiantes** pueden inscribirse en varias **especialidades** y en diversos **cursos**. El estudiante con clave **S01** tiene su **especialidad** en **sistemas** y **Bioquímica** y toma los cursos de

Natación y **Danza**, el estudiante **B01** tiene la especialidad en **Bioquímica** y toma el curso de **Guitarra**, el estudiante con clave **C03** tiene la especialidad de **Civil** y toma el curso de **Natación**.

En esta tabla o relación **no existe dependencia funcional** porque los estudiantes pueden tener distintas especialidades, un valor único de clave puede poseer muchos valores de especialidades al igual que de valores de cursos. Por lo tanto existe **dependencia de valores múltiples**. Este tipo de dependencias produce redundancia de datos, como se puede apreciar en la tabla anterior, en donde la clave **S01** tiene tres registros para mantener la serie de datos en forma independiente lo cual ocasiona que al realizarse una actualización se requiera de **demasiadas operaciones para tal fin**.

Existe una **dependencia de valores múltiples** *cuando una afinidad tiene por lo menos tres atributos, dos de los cuales poseen valores múltiples y sus valores dependen solo del tercer atributo*, en otras palabras en la afinidad **R (A,B,C)** existe una dependencia de valores múltiples si **A determina valores múltiples de B**, **A determina valores múltiples de C** , y **B y C son independientes entre sí**.

En la tabla anterior **Clave** determina valores múltiples de **especialidad** y **clave** determina valores múltiples de **curso,** pero **especialidad** y **curso** son independientes entre sí.

Las dependencias de valores múltiples se definen de la siguiente manera:

Clave ->->Especialidad y Clave->->Curso; Esto se lee "Clave multidetrmina a Especialidad, y clave multidetermina a Curso"

Para eliminar la redundancia de los datos, se deben eliminar las **dependencias de valores múltiples**. Esto se logra construyendo **dos** tablas, donde cada una almacena datos para solamente uno de los atributos de valores múltiples.

Para nuestro ejemplo, las tablas correspondientes son:

Tabla Especialidad

Clave	Especialidad
S01	Sistemas
S01	Bioquimica
B01	Bioquimica
C03	Civil

Tabla Curso

Clave	Curso
S01	Natación
S01	Danza
B01	Guitarra
C03	Natación

Quinta forma normal

Un esquema de relaciones R está en 5FN con respecto a un conjunto D de dependencias funcionales, de valores múltiples y de producto, si para todas las dependencias de productos en D se cumple por lo menos una de estas condiciones:

- ▶ (R1, R2, R3, ... Rn) es una dependencia de producto trivial.
- ▶ Toda Ri es una superllave de R.

La quinta forma normal se refiere a dependencias que son extrañas. Tiene que ver con tablas que pueden dividirse en subtablas, pero que no pueden reconstruirse. Recuerda lo que haciamos cuando teniamos entidades que tenían **especializaciones o roles**.

Nota: Para comprender mejor este tema es necesario que leas el capítulo 5 del libro **"Database processing fundamentals, design and implementation"**, ya que en él se explica a detalle los diferentes tipos de anomalias que evitamos al normalizar una base de datos, aspecto que en la lectura anterior no se detallan a profundidad.

KROENKE, David M., Database processing fundamentals, design and implementation. 6a. edición. Upper Saddle River, New Jersey, U.S.A. Prentice Hall, Inc. 1998. 523p. ISBN 0-13-737842-4.

Este libro se encuentra en la biblioteca.

Powered by Materialize.