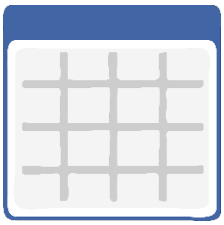
TEMA 6
NORMALIZACIÓN

6.1. PROCESO DE NORMALIZACIÓN.

Normalización: Proceso empleado para evaluar y corregir la estructura de tablas con la finalidad de minimizar redundancia reduciendo así la posibilidad de existencia de anomalías en los datos.
El proceso de normalización se desarrolla a través de la aplicación de una serie de pasos o estados llamados formas normales:

- Primera forma normal (1FN).
- Segunda forma normal (2FN)
- Tercera forma normal (3FN)
- Forma normal de Boyce –Codd (BCNF)
- Cuarta forma normal (4FN).



La aplicación de un mayor o menor nivel de normalización puede generar las 2 siguientes situaciones:

Mayor nivel de Normalización
- Posibles problemas de desempeño y eficiencia
- Menor o prácticamente ausencia de redundancia de datos.

Menor nivel de Normalización
- Prácticamente sin problemas de desempeño y eficiencia.
- Mayor nivel de redundancia de datos.

- Típicamente, para la mayoría de los modelos, se emplea hasta la 3FN.
- Desde el punto de vista estructural, una forma normal de mayor nivel es mejor que una de menor nivel.
- El proceso de normalización es importante pero un alto nivel ***no siempre es lo deseado o lo mejor***.
- A mayor nivel de normalización la información se almacena en un mayor número de tablas y por lo tanto se incrementará el número de operaciones `join` entre ellas para satisfacer una consulta.
- Antes de decidir el nivel de normalización a aplicar se deberán considerar aspectos ***de rapidez y desempeño***.
- Al proceso inverso de normalización se le conoce como ***denormalización.***

Para realizar el proceso de normalización, considerar la siguiente tabla de datos en la que se almacenan las faltas y las calificaciones de los alumnos de una universidad, en donde cada alumno pertenece a una carrera la cual también se especifica en la tabla de datos.

num_ estudiante	nombre	ap_ paterno	ap_ materno	clave_ asignatura	nombre_ asignatura	créditos_ asignatura	faltas	calificación	clave_ nacimiento	Lugar_ nacimiento	clave_ carrera	nombre_ carrera
1001	Juan	Méndez	Kim	1763	Algebra	10	1	9	COL	Colima	110	I. Civil
				3411	Calculo 2	8	0	7	COL	Colima	110	I. Civil
1002	Mario	Luna	Ubaldo	1890	Calculo 1	10	3	7	CHIH	Chihuahua	110	I. Civil
1003	Eva	Aguirre	Salas	3411	Calculo 2	8	5	8	NL	Nuevo León	111	I. Electro
1004	Lucia	Juárez	Aldama	1763	Algebra	10	0	10	MICH	Michoacán	111	I. Electro
1005	Alonso	Lugo	López	1890	Calculo 1	10	2	5	SON	Sonora	111	I. Electro
1002	Mario	Luna	Ubaldo	1763	Algebra	10	2	8	CHIH	Chihuahua	110	I. Civil
1006	Eva	Lugo	Macías	1790	Estadística	8	9	6	QRO	Querétaro	111	I. Electro

¿Qué anomalías existen en esta tabla de datos que podrían ser eliminadas al aplicar un proceso de normalización?

- Al existir varios registros por alumno, su información se repite en los campos `nombre`, `ap_paterno`, `ap_materno`, `clave_nacimiento`, `lugar_nacimiento`, `clave_carrera` y `nombre_carrera`.
- Los nombres de las asignaturas en el campo `nombre_asignatura` se repiten por alumno, lo mismo ocurre para los campos `lugar_nacimiento`, `nombre_carrera`.
- En caso de aplicar una actualización, si no se actualizan los campos con valores repetidos, genera inconsistencia (*anomalías de actualización*).
- ¿Qué pasaría si se registra a un alumno que no está inscrito en alguna asignatura? Todos los demás campos deberán declararse como `null` (*anomalías de inserción*).
- ¿Qué pasa si se elimina el registro del alumno 1006?, Los datos de la asignatura estadística desaparecen ya que solo este alumno hace referencia a dicha asignatura (*anomalías de eliminación*).

6.2. APLICACIÓN DE LA 1FN

Objetivo:
Una tabla estará en su 1FN cuando no existen ***grupos de repetición***, la PK, ***dependencias parciales*** y ***dependencias transitivas*** se han identificado.

6.2.1. Grupos de repetición.

Ocurre al realizar una agrupación de 2 o más registros para una o varias columnas que tienen el mismo valor. Por ejemplo, para la tabla de datos anterior, los primeros 2 registros son agrupados en los campos nombre, `ap_paterno` y `ap_materno`. En la 1FN, estas agrupaciones se deberán eliminar repitiendo los datos en cada registro:

num_ estudiante	nombre	ap_ paterno	ap_ materno	clave_ asignatura	nombre_ asignatura	créditos_ asignatura	faltas	calificación	clave_ nacimiento	Lugar_ nacimiento	clave_ carrera	nombre_ carrera
1001	Juan	Méndez	Kim	1763	Algebra	10	1	9	COL	Colima	110	I. Civil
1001	**Juan**	**Méndez**	**Kim**	3411	Calculo 2	8	0	7	COL	Colima	110	I. Civil

6.2.2. Dependencia funcional.

Recordando este concepto visto en el tema 3:
Consiste en determinar y verificar el o los campos que actuarán o tendrán el papel de PK.
Definición: El atributo “B” es funcionalmente dependiente de un atributo “A” si cada valor de la columna “A” determina uno y solo un valor de la columna “B”. Se expresa:

$$A \rightarrow B$$

De lo anterior, se puede concluir que “A” puede tomar el rol de llave primaria, siempre y cuando A, también pueda determinar el valor de las demás columnas. Es decir:

$$A \rightarrow B, C, D, E \ldots$$

Ejemplo:

Considerando los campos de la tabla anterior, iniciando en C_1, C_2, hasta C_{13} de izquierda a derecha, determinar si las siguientes expresiones son verdaderas

- ¿C_4 -> C_3? (¿C_3 será funcionalmente dependiente de C_4?), Respuesta: FALSE.
- ¿C_4 ->C_1? (¿C_1 será funcionalmente dependiente de C_4?), Respuesta: FALSE
- ¿C_1->C_4? (¿C_4 será funcionalmente dependiente de C_1?), Respuesta: TRUE
- ¿C_5->C_7? (¿C_7 será funcionalmente dependiente de C_5?), Respuesta: TRUE
- ¿C_1->C_8? Respuesta FALSE
- ¿C_5->C_8? Respuesta: FALSE
- ¿C_1, C_5->C_8, C_9? Respuesta: TRUE

6.2.3. Identificación de la PK.

Empleando el concepto de ***dependencia funcional*** se determina la PK. El proceso consiste en encontrar el(los) campo(s) que determinen de manera única a cada uno de los registros de la tabla:

A -> B

- Todos los atributos de la tabla representados por "B" son conocidos como ***atributos dependientes*** de A.
- Todos los atributos de la tabla representados por "A" son conocidos como ***atributos determinantes*** debido a que en su conjunto pueden identificar o determinar de manera única a cada registro de la tabla. En este sentido, todos los atributos formados por A representan a la PK de la tabla.

Para la tabla anterior, la PK estará definida por los siguientes atributos:

num_estudiante, clave_asignatura ->
nombre, ap_paterno, ap_materno, nombre_asignatura, créditos_asignatura, faltas, calificación, clave_nacimiento, lugar_nacimiento, clave_carrera, nombre_carrera.

A través de la combinación de ambos atributos, es posible determinar de forma única a los demás.

6.2.4. Dependencias parciales.

- La dependencia parcial puede existir únicamente en tablas con una llave primaria compuesta.
- En una dependencia parcial, alguno de los atributos que forma parte de la PK puede por si solo actuar como ***atributo determinante*** de uno o más atributos de la tabla.

Ejemplo:

Para la tabla de datos, la primera condición se cumple, existe una PK compuesta. Para verificar si existen dependencias parciales, se verifica si los campos que integran a la PK de forma individual pueden determinar a otros campos:

num_estudiante ->
nombre, ap_paterno, ap_materno, clave_nacimiento, lugar_nacimiento, clave_carrera, nombre_carrera

clave_asignatura -> nombre_asignatura, créditos_asignatura

- Observar que existen 2 dependencias parciales. A partir del número de estudiante se pueden determinar los campos que aparecen del lado derecho, y lo mismo sucede con la clave de la asignatura.
- Observar que podría ocurrir lo mismo con la clave de la carrera, sin embargo, esta no es una dependencia parcial, ya que el atributo determinante debe formar parte de la PK.

6.2.5. Dependencias transitivas.

- Una dependencia transitiva ocurre cuando se detecta una dependencia funcional en donde el atributo determinante no forma parte de la PK (a diferencia de las dependencias parciales).
- En este caso, no importa si la tabla tiene PK simple o compuesta.

Ejemplo:

Para la tabla de datos mostrada anteriormente, se tienen las siguientes relaciones transitivas:

clave_nacimiento -> lugar_nacimiento

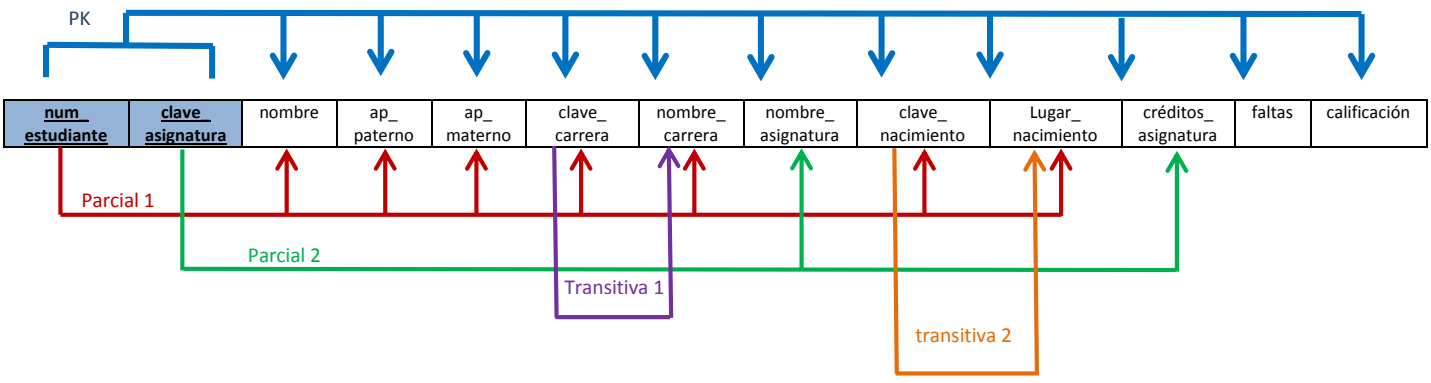
clave_carrera -> nombre_carrera

Recordando, para que una tabla este en 1FN:

- Paso 1: Se identifica la PK
- Paso 2: Se eliminan grupos de repetición
- Paso 3: Se identifican dependencias parciales.
- Paso 4: Se identifican dependencias transitivas.

Este resultado se describe en los llamados diagramas ***de dependencias***:

6.2.6. Diagrama de dependencias para la 1F



6.3. APLICACIÓN DE LA 2FN

Objetivo:

Una tabla está en su 2FN cuando:

- La tabla está en su 1FN
- Se han eliminado las dependencias parciales.

Si la tabla no tiene una PK compuesta, en automático esta se encuentra en su 2FN.

6.3.1. Paso 1: Eliminación de dependencias parciales.

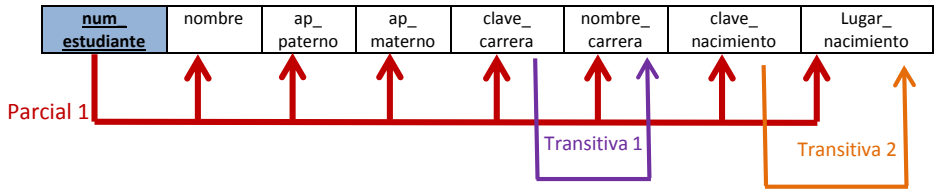
- Por cada dependencia parcial identificada se crea una tabla nueva asignando como PK al atributo determinante y con sus atributos dependientes como atributos de la tabla.
- Se le asigna un nombre a la tabla nueva.
- De la tabla original en su 1FN se eliminan únicamente los atributos ***dependientes*** que integran a las nuevas tablas.

Ejemplo:

Para la tabla de datos, se tienen 2 nuevas tablas:

- Nombre de la tabla: `estudiante`.

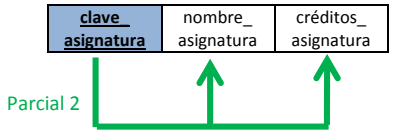
`num_estudiante ->`
`nombre, ap_paterno, ap_materno, clave_nacimiento, lugar_nacimiento, clave_carrera, nombre_carrera`



Observar que las relaciones transitivas permanecen aún.

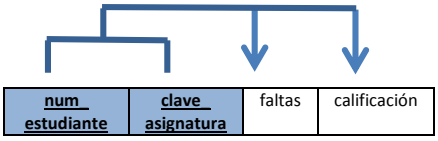
- Nombre de la tabla: `asignatura`.

`clave_asignatura ->` `nombre_asignatura, créditos_asignatura`



- Tabla Original: `estudiante_asignatura`

`num_estudiante, clave_asignatura ->` `faltas, calificación`



- Observar que ahora la tabla original solo contiene los campos que dependen de la PK compuesta.
- Observar que las PKs de las tablas nuevas, son ahora PKs y FKs de la tabla original.
- Hasta el momento se tienen 3 tablas:
  - `estudiante, asignatura, estudiante_asignatura`

6.4. APLICACIÓN DE LA 3FN.

Objetivo.
Una tabla está en su 3FN cuando:
- La tabla está en su 2FN
- Se han eliminado las dependencias transitivas.

6.4.1. Eliminación de dependencias transitivas.

- El procedimiento es similar al de la 2FN. Para cada dependencia transitiva se crea una tabla nueva.
- De la tabla original en su 2FN se eliminan únicamente los atributos dependientes que integran a las nuevas tablas.

Ejemplo:
- La tabla `asignatura` se queda en su 2FN ya que no cuenta con dependencias transitivas.
- Para la tabla `estudiante`, que es la que contiene las 2 dependencias transitivas se tiene:

- Nombre de la tabla: `carrera`.

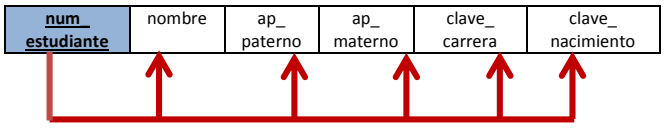
`clave_carrera ->` `nombre_carrera`

- Nombre de la tabla: `lugar_nacimiento`.

`clave_nacimiento ->` `lugar_nacimiento`

- Tabla original: `estudiante`

`num_estudiante ->`
`nombre, ap_paterno, ap_materno, clave_nacimiento, clave_carrera`

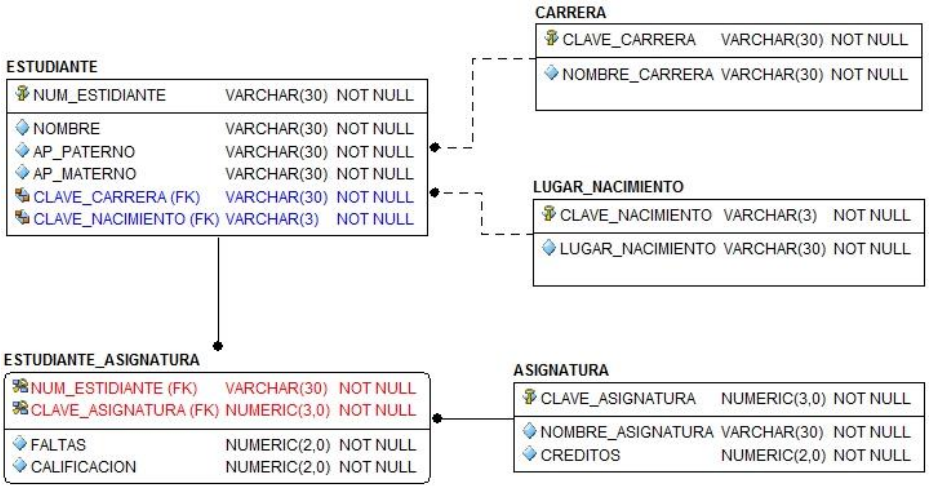


- Observar nuevamente, los campos `clave_carrera` y `clave_nacimiento`, ahora son FKs de las tablas `carrera` y `lugar_nacimiento` respectivamente.
- En tercera forma normal se tienen las siguientes tablas:

- estudiante, asignatura, estudiante_asignatura, carrera, lugar_nacimiento.

6.4.2. Construcción del modelo relacional.

Con base a los diagramas de dependencias que se obtuvieron durante el proceso anterior, es posible generar un modelo relacional:



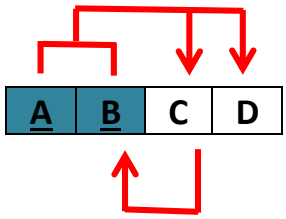
- Observar que el diagrama se genera con las tablas y atributos que se determinaron en el proceso de normalización.
- Un siguiente paso consiste en realizar el refinamiento del diseño, en el que se pueden aplicar algunas mejoras. Para el ejemplo se pueden realizar los siguientes campos:
  - Agregar una PK artificial numérica como PK para mejorar desempeño y/o evitar el problema de posibilidad de cambios en las PKs. Hay que recordar que las PKs que se determinan en este proceso son llaves primarias naturales (campos que forman parte de las reglas de negocio).
  - Agregar una PK artificial a la tabla intermedia (similar a la técnica que se revisó en el capítulo 4).
- Observar que la tabla original de datos fue fragmentada en 5 tablas, y al final del proceso, esta tabla resultó ser una tabla intermedia que implementa una relación M:N entre `estudiante` y `asignatura`.

6.5. APLICACIÓN DE FORMAS NORMALES DE ORDEN SUPERIOR.

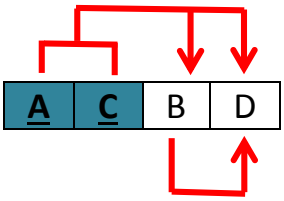
6.5.1. Forma normal de Boyce – Codd

Esta forma normal es una variante de la 3FN, y fue desarrollada en 1974 por Raymond F. Boyce y Edgar F. Codd en la cual se resuelven algunas anomalías que no resuelve la 3FN.

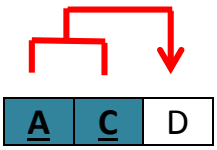
- Ocurre en tablas con PK compuesta en donde existen atributos que son determinantes y que por alguna razón no fueron seleccionados para formar parte de la PK. A nivel general se tiene la siguiente condición:

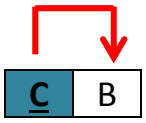


- Observar que el campo C determina al campo B, pero el campo C no es parte de la PK.
- Dado que C->B, resulta completamente válido modificar la PK de la tabla intercambiando a B con C:



- La PK sigue cumpliendo con su función ya que C es un atributo determinante.
- Observar que al hacer el intercambio de estos s atributos, se forma una dependencia parcial: C->B
- Recordando el proceso para normalizar una tabla en su 2FN, la tabla en BCNF ocurre al eliminar la dependencia parcial que se acaba de formar. Por lo tanto, la tabla en su BCNF se divide en 2:





Las 2 tablas anteriores ahora es tan en su forma normal Boyce – Codd (BCNF)

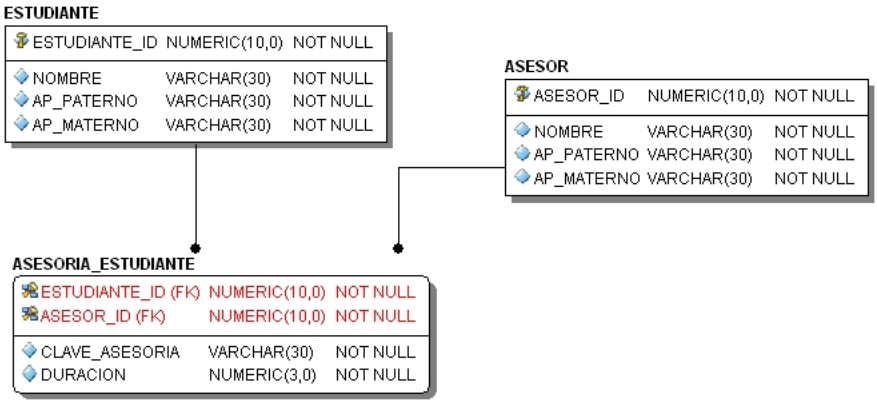
Para comprender el concepto y las anomalías que se generan al no realizar este proceso se considerar el siguiente escenario:

Ejemplo:

Registro de las asesorías y de los profesores asignados a un alumno.

- Un estudiante puede tener a varios asesores asignados durante un semestre y un asesor puede tener asignados a varios alumnos.
- Un asesor puede dar varias asesorías. Cada asesoría se identifica por una clave. Una asesoría la ofrece un solo asesor.
- El estudiante puede solicitar asesoría por parte de uno de sus asesores y al acudir se guarda el tiempo que dura dicha asesoría.

El modelo relacional que se ha generado para modelar estas reglas de negocio es el siguiente:



La siguiente tabla muestra el comportamiento de los datos para la tabla `asesoria_estudiante`

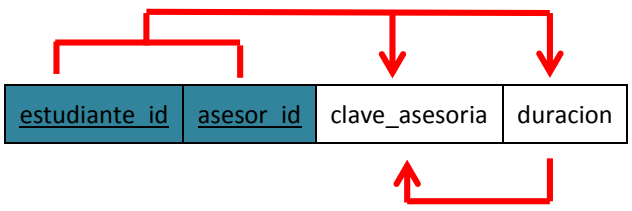
estudiante_id	asesor_id	clave_asesoria	duración (min)
1	1	ALG-001	15
2	1	ALG-001	22
3	2	BD-022	19
4	3	MATH-089	34
4	4	CALC-020	32
5	4	CALC-020	27
6	5	JAVA-093	21

¿Qué anomalías presenta este diseño?

- La tabla `asesoria_estudiante` intenta describir 2 reglas:
  - Las asesorías a las que acudió el estudiante.
  - Las asesorías asignadas a un asesor.
- Lo anterior tiene como resultado anomalías de actualización y de eliminación:
  - Anomalía de actualización: Si al asesor con Id = 1 se le cambia su asesoría a ALG-020, se deberán actualizar 2 registros en la tabla, la asignación de la asesoría para el asesor 1 se duplica.
  - Anomalía de eliminación: Si se elimina el registro del estudiante con Id =6, ¿cómo sabríamos la asesoría asignada el profesor con Id = 5?

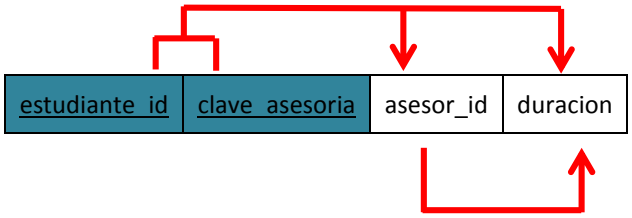
Observar que el atributo `clave_asesoria` es un atributo determinante de `asesor_id`, es decir, a partir de la clave de la asesoría es posible determinar al asesor que la ofrece.

`clave_asesoria` `-> asesor_id`

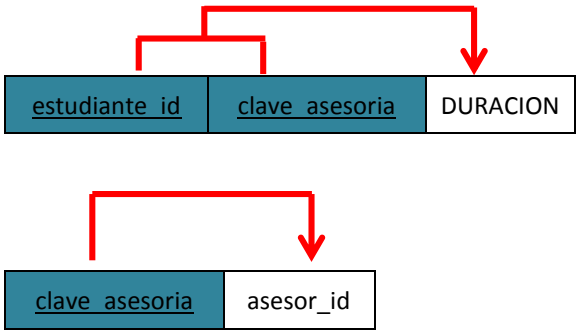


Si se realizara el intercambio para generar una dependencia parcial, la tabla quedaría de esta forma:

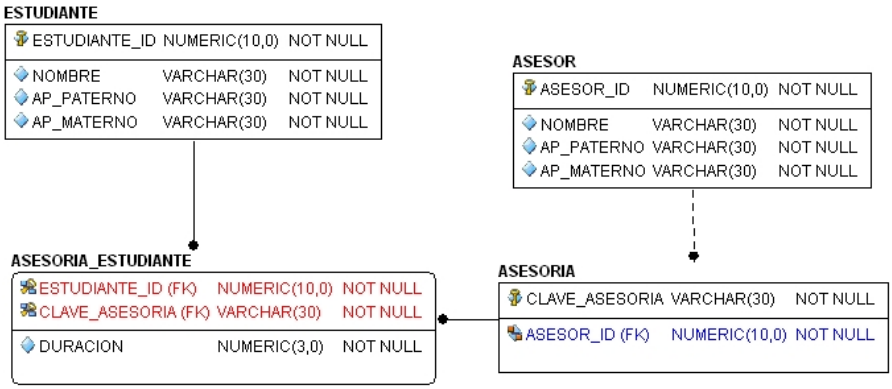
`estudiante_id,clave_asesoria` `-> asesor_id,duracion`



Observar que la tabla sigue siendo equivalente a la original, pero ahora con una dependencia parcial, la cual al eliminarla se tiene a la tabla en su BCNF:



El modelo relacional resultante será:



Con este nivel de normalización la tabla `asesoria_estudiante` ahora solo describe las asesorías a las que acude un estudiante. La definición de las asesorías que ofrece cada asesor se encuentra en la tabla nueva `asesoria`.

6.5.2. Aplicación de la cuarta forma normal (4FN)

Una tabla estará en su cuarta forma normal (4FN) cuando:

- Está en su tercera forma normal 3FN
- Se han eliminado posibles **dependencias multivalor**.

6.5.2.1. Depedencias multivalor

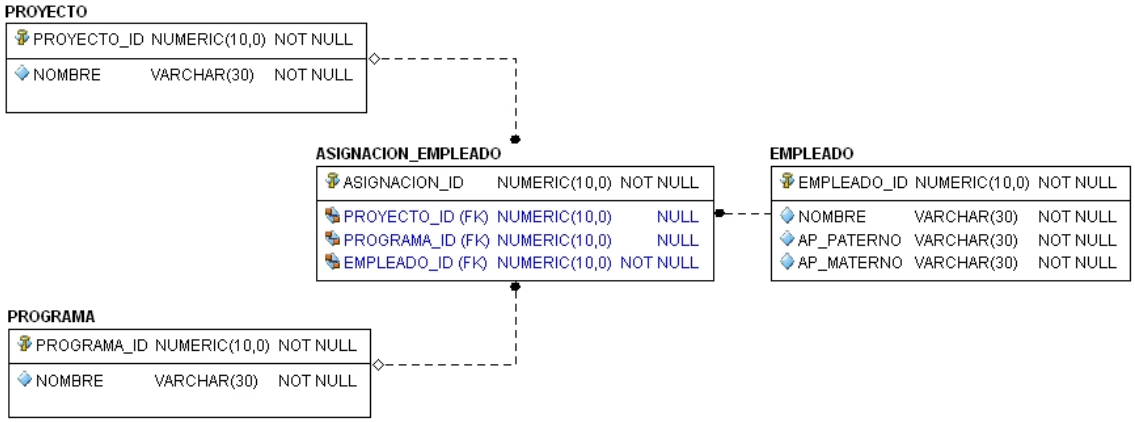
Ocurre cuando la PK de una tabla determina múltiples valores de 2 o más campos de la tabla y cada uno de estos campos son totalmente independientes entre sí, es decir, no hay relación alguna entre ellos.

Ejemplo:

Programas y proyectos en los que puede estar inscrito un empleado:

- Un empleado puede estar asignado a uno o a mas proyectos
- Un proyecto está formado por varios empleados.
- Un empleado puede participar en uno o más programas voluntarios de apoyo.
- Un programa de apoyo está integrado por varios empleados.

El modelo relacional que se ha generado para modelar estas reglas de negocio es el siguiente:



La siguiente tabla muestra el comportamiento de los datos para la tabla `asignacion_empleado`.

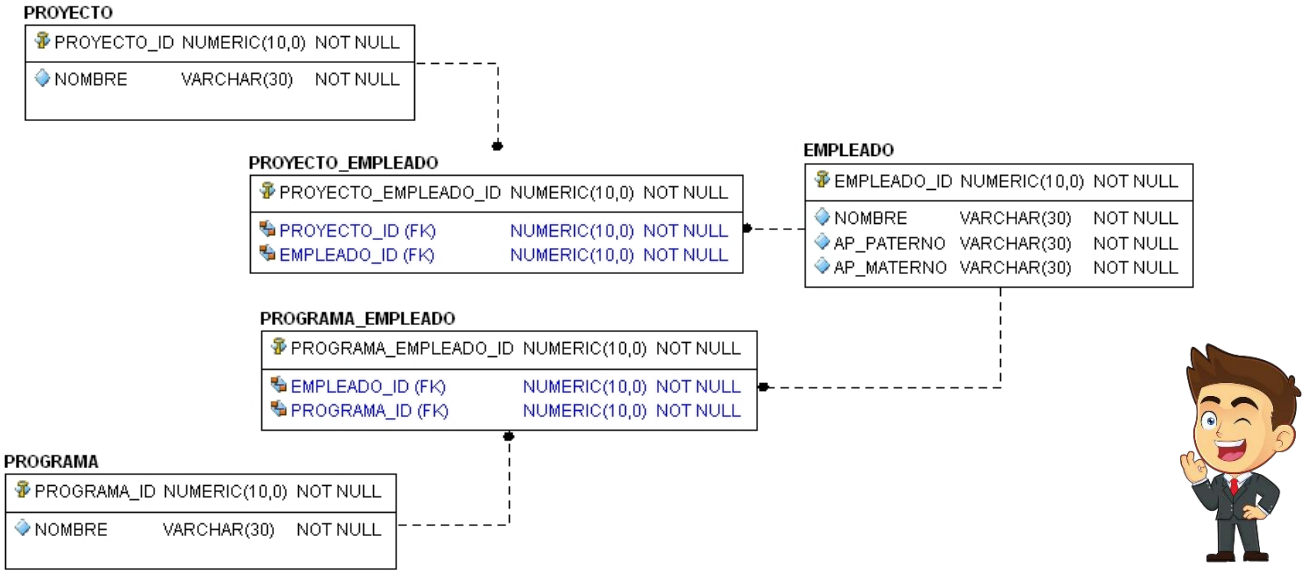
asignacion_id	proyecto_id	programa_id	empleado_id
1	4	NULL	10
2	5	NULL	10
3	NULL	2	10
4	NULL	3	10

¿Qué anomalías presenta este diseño?

- Observar la dependencia multivalor: `proyecto_id` y `programa_id` son 2 campos totalmente independientes, y sin embargo están asociados a un mismo empleado.
- Para un mismo empleado se pueden generar múltiples valores de `proyecto_id` y `programa_id`. En otras palabras, estas anomalías se pueden generar cuando se emplea una misma tabla intermedia para representar más de una relación M:N, en este caso son 2 relaciones M:N `empleado – proyecto`, y `empleado – programa`.
- Estas 2 condiciones provocan que ambos campos se tengan que definir como NULL como se muestra en la tabla de datos.

Para eliminar la dependencia multivalor, se crea una tabla nueva por cada campo que genera múltiples valores, es decir, una tabla por cada relación M:N

El diagrama en su 4FN es el siguiente:



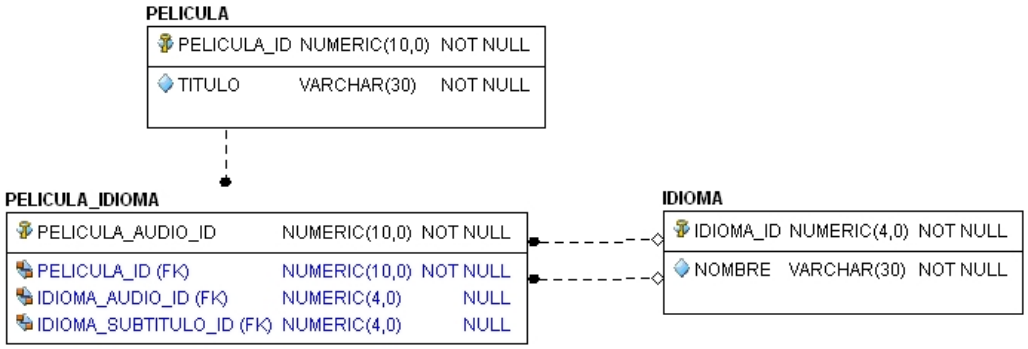
Ejemplo:

Registro de películas:

- Una película tiene varios idiomas de audio disponibles. Un idioma puede estar asociado a varias películas.
- Una película tiene varios idiomas de subtítulos disponibles. Un idioma puede estar asociado a varias películas.

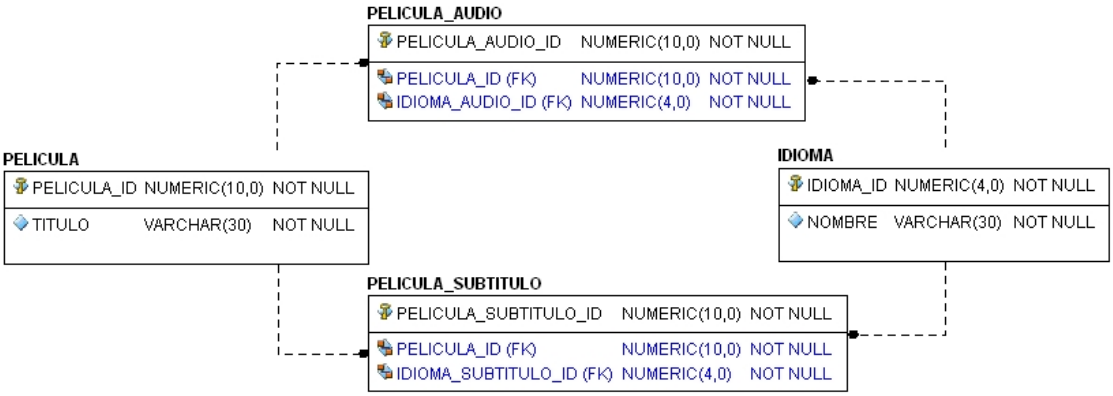
Como se puede observar, se trata de 2 relaciones M:N. Los subtítulos y los audios en sentido estricto son independientes, es decir, se pueden seleccionar de forma independiente, por lo tanto, es posible generar 2 tablas intermedias, una para cada relación M:N (4FN).

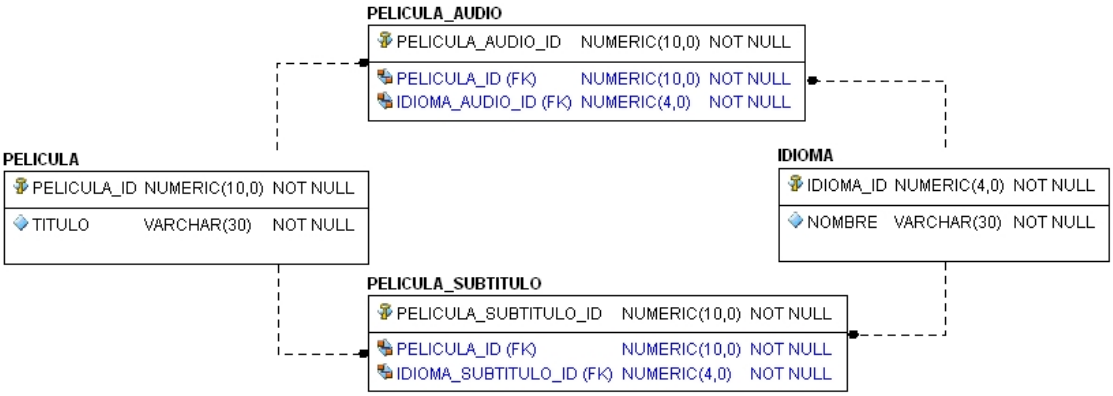
Sin este nivel de normalización, se tendrían los siguientes inconvenientes:



asignacion_idioma_id	idioma_audio_id	idioma_subtitulo_id	película_id
	NULL	1	1
	NULL	2	1
	3	NULL	1
	4	NULL	1

- Observar que se deben definir los campos como NULL, o en su defecto, se tendrían que agregar todas las posibles combinaciones entre idiomas de audio y de subtítulos. Lo anterior obliga a relacionar los idiomas de los audios con los idiomas de los subtítulos cuando en sentido estricto son independientes.
- Aplicando la 4FN, el modelo relacional es el siguiente:


PELICULA_AUDIO
PELICULA_AUDIO_ID NUMERIC(10,0) NOT NULL
PELICULA_ID (FK) NUMERIC(10,0) NOT NULL
IDIOMA_AUDIO_ID (FK) NUMERIC(4,0) NOT NULL
PELICULA
PELICULA_ID NUMERIC(10,0) NOT NULL
TITULO VARCHAR(30) NOT NULL
IDIOMA
IDIOMA_ID NUMERIC(4,0) NOT NULL
NOMBRE VARCHAR(30) NOT NULL
PELICULA_SUBTITULO
PELICULA_SUBTITULO_ID NUMERIC(10,0) NOT NULL
PELICULA_ID (FK) NUMERIC(10,0) NOT NULL
IDIOMA_SUBTITULO_ID (FK) NUMERIC(4,0) NOT NULL


PELICULA_AUDIO
PELICULA_AUDIO_ID NUMERIC(10,0) NOT NULL
PELICULA_ID (FK) NUMERIC(10,0) NOT NULL
IDIOMA_AUDIO_ID (FK) NUMERIC(4,0) NOT NULL
PELICULA
PELICULA_ID NUMERIC(10,0) NOT NULL
TITULO VARCHAR(30) NOT NULL
IDIOMA
IDIOMA_ID NUMERIC(4,0) NOT NULL
NOMBRE VARCHAR(30) NOT NULL
PELICULA_SUBTITULO
PELICULA_SUBTITULO_ID NUMERIC(10,0) NOT NULL
PELICULA_ID (FK) NUMERIC(10,0) NOT NULL
IDIOMA_SUBTITULO_ID (FK) NUMERIC(4,0) NOT NULL