Clase 7 – Proceso de Normalización

Lo que vamos a ver hoy:

- *Normalización*
- *Normalización de atributos*
- *Dependencias Funcionales*
- *Reglas de Derivación*
- *Significado de los atributos en una relación (Cardinalidad)*
- *Formas Normales*

NORMALIZACIÓN

Base de Datos I

Lic. Leandro Mariano Abraham

Clase 7 – Normalización

La normalización es el proceso de organizar los datos de una base de datos. Se incluye la creación de tablas y el establecimiento de relaciones entre ellas según reglas diseñadas tanto para proteger los datos como para hacer que la base de datos sea más flexible al eliminar la redundancia y las dependencias incoherentes.

Ejemplo de Normalización:

Factura

Nro	Fecha	CUIT	Nombre	Telefono	Importe
1	12/01/2020	11111111	Juan Perez	1144552445	1000
2	12/01/2020	2222222	María Sanchez	1156467254	2000
3	13/01/2020	3333333	Javier Rodriguez	1134227685	2000
4	13/01/2020	11111111	Juan Perez	1144552445	1500
5	14/01/2020	11111111	Juan Perez	1144552445	4000

Problemas de este modelo:

- *Redundancia*
- *Inconsistencia*

Mediante la normalización buscamos obtener un esquema de base de datos con redundancia mínima y sin inconsistencias

Ejemplo de Normalización:

Nro	Fecha	CUIT	Nombre	Telefono	Importe
1	12/01/2020	11111111	Juan Perez	1144552445	1000
2	12/01/2020	2222222	María Sanchez	1156467254	2000
3	13/01/2020	3333333	Javier Rodriguez	1134227685	2000
4	13/01/2020	11111111	Juan Perez	1144552445	1500
5	14/01/2020	11111111	Juan Perez	1144552445	4000

CUIT → Nombre
CUIT → Telefono

Dependencias Funcionales

Esto se debe obtener

Factura

Nro	Fecha	CUIT	Importe
1	12/01/2020	11111111	1000
2	12/01/2020	2222222	2000
3	13/01/2020	3333333	2000
4	13/01/2020	11111111	1500
5	14/01/2020	11111111	4000

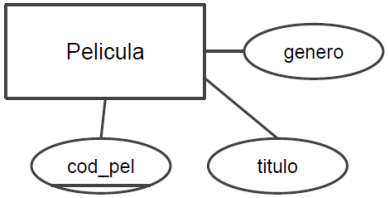
Cliente

CUIT	Nombre	Telefono
11111111	Juan Perez	1144552445
2222222	María Sanchez	1156467254
3333333	Javier Rodriguez	1134227685

Proceso de Normalización

NORMALIZACIÓN DE ATRIBUTOS

Ejemplo de Normalización de atributos:



Pelicula

cod_pel	titulo	genero
1000	Volver al futuro	C. Ficción
1001	Volver al futuro 2	Ciencia Ficción
1002	La llamada	Terror
1003	Duro de matar	Acción
1004	Duro de matar 2	Accion
1005	Duro de matar 3	Axión

¿Qué problemas podemos observar que devienen de registrar de esta manera el género?

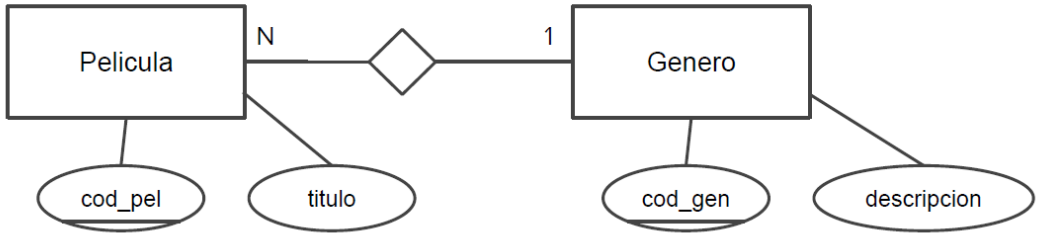
Proceso de Normalización:

Pelicula

cod_pel	titulo	cod_gen
1000	Volver al futuro	1
1001	Volver al futuro 2	1
1002	La llamada	2
1003	Duro de matar	3
1004	Duro de matar 2	3
1005	Duro de matar 3	3

Genero

cod_gen	descripcion
1	Ciencia Ficción
2	Terror
3	Acción



Campos claves dentro de las tablas:

- ***Clave Primaria:*** *Identifica de manera univoca a cada uno de los registros de la tabla. No puede ni repetirse ni ser nula.*

  EJ: Película → cod_pel
  Genero → cod_gen

- ***Clave foránea*** *(Segundaria o ajena) : Es el campo que me une con otra tabla. Generalmente es un campo que es clave primaria en la otra tabla.*

DEPENDENCIAS FUNCIONALES

Base de Datos I

Lic. Leandro Mariano Abraham

Clase 7 – Normalización

Las dependencias funcionales son un concepto fundamental en la teoría de bases de datos relacionales. Este concepto se utiliza para describir la relación entre diferentes atributos en una tabla y es crucial para el diseño y la normalización de bases de datos. Existen diferentes tipos de dependencias funcionales en bases de datos, las cuales abordaremos a continuación.

Dependencia Funcional Completa

Una dependencia funcional X→Y es completa si al eliminar cualquier atributo de X, la dependencia ya no se mantiene.

Tabla Prestamos de Biblioteca

ID_Prestamo	ID_Libro	ID_Usuario	Fecha_Prestamo	Fecha_Devolucion
1	L001	U001	1/1/2024	15/1/2024
2	L002	U002	3/1/2024	17/1/2024
3	L003	U003	5/1/2024	19/1/2024
4	L001	U004	7/1/2024	21/1/2024

- *(ID_Libro, ID_Usuario) → Fecha_Prestamo*
- *(ID_Libro, ID_Usuario) → Fecha_Devolucion*

La combinación de ID_Libro y ID_Usuario determina la Fecha_Prestamo y la Fecha_Devolucion. Es decir, dado un libro y un usuario específico, podemos determinar de manera única las fechas de préstamo y devolución.

Comprobación

Para verificar que esta dependencia es completa, consideremos eliminar uno de los atributos de la combinación:

- ***ID_Libro*** → *Fecha_Prestamo: No es cierto, ya que un mismo libro puede ser prestado en diferentes fechas a diferentes usuarios.*

- ***ID_Usuario*** → *Fecha_Prestamo: No es cierto, ya que un mismo usuario puede prestar diferentes libros en diferentes fechas.*

*Por lo tanto, necesitamos ambos atributos (**ID_Libro** y **ID_Usuario**) para determinar de manera única las fechas de préstamo y devolución. Esto confirma que la dependencia funcional es completa.*

Cabe aclarar, además, que existe una dependencia funcional entre el ID_préstamo y el resto de los atributos de la tabla.

- *ID_Prestamo → ID_Libro, ID_Usuario, Fecha_de_Prestamo, Fecha_de_Devolución*

Dependencia Funcional Parcial

Una dependencia funcional X→Y es parcial si un subconjunto propio de X aún determina Y. En otras palabras, se puede eliminar uno o más atributos de X y la dependencia aún se mantiene.

Tabla Ventas de Tiendas

ID_Venta	ID_Producto	ID_Tienda	Precio_Unitario	Cantidad	Total_Venta
1	P001	T001	10.00	2	20.00
2	P002	T001	15.00	1	15.00
3	P001	T002	10.50	3	31.50
4	P003	T001	20.00	1	20.00

- ***(ID_Producto, ID_Tienda) → Precio_Unitario***

Esta es una dependencia funcional que no es parcial porque necesitamos ambos atributos, ID_Producto e ID_Tienda, para determinar el Precio_Unitario. No podemos eliminar ninguno de ellos.

- ***(ID_Producto, ID_Tienda, Cantidad) → Total_Venta***

Esta dependencia es una dependencia funcional completa porque necesitamos los tres atributos para determinar el Total_Venta.

Ejemplo de Dependencia Funcional Parcial

Ahora, para entender mejor una dependencia funcional parcial, supongamos que:

(ID_Producto, ID_Tienda) determina Precio_Unitario, es decir, (IDProducto,IDTienda) → PrecioUnitario

Si eliminamos **Cantidad**, aún tenemos una dependencia funcional porque ***(IDProducto,IDTienda) → PrecioUnitario.***

Esto muestra que **(IDProducto,IDTienda)** es suficiente para determinar **Precio_Unitario.** Por lo tanto, la dependencia **(IDProducto,IDTienda,Cantidad)** → ***PrecioUnitario*** es parcial en relación con **(IDProducto,IDTienda)** porque podemos eliminar **Cantidad** y aún tener una dependencia funcional válida.

Dependencia Funcional Transitiva

Una dependencia funcional transitiva en una base de datos ocurre cuando un atributo A determina un atributo B, y B determina un atributo C, entonces A también determina C de manera indirecta a través de B. En otras palabras, si A → B y B → C, entonces A → C.

Tabla de Libros

ID_Libro	Titulo	ID_Autor	Nombre_Autor	ID_Editorial	Nombre_Editorial
1	El Quijote	A001	Miguel de Cervantes	E001	Editorial A
2	Cien Años de Soledad	A002	Gabriel García Márquez	E002	Editorial B
3	Don Juan Tenorio	A003	José Zorrilla	E001	Editorial A
4	La Sombra del Viento	A004	Carlos Ruiz Zafón	E003	Editorial C

En esta tabla, tenemos varias dependencias funcionales:

- ***ID_Libro → ID_Autor***
- ***ID_Autor → Nombre_Autor***

Dado que ID_Libro determina ID_Autor y ID_Autor determina Nombre_Autor, podemos concluir que:

- ***ID_Libro → Nombre_Autor***

Esto representa una dependencia funcional transitiva porque ID_Libro determina Nombre_Autor de manera indirecta a través de ID_Autor

La dependencia funcional transitiva es importante porque su existencia puede indicar que la base de datos no está completamente normalizada. Identificar y eliminar estas dependencias es esencial para lograr un diseño de base de datos más eficiente y evitar problemas de redundancia y actualización.

Dependencia Funcional Multivaluada

Una dependencia funcional multivaluada ocurre cuando un atributo en una tabla determina un conjunto de valores, en lugar de un solo valor.

Tabla de Habilidades y Proyectos

Proyecto_ID	Empleado_ID	Habilidad
101	E01	Java
101	E01	Python
102	E02	SQL

ProyectoID, EmpleadoID ↠ Habilidad*. Para cada combinación de* ***Proyecto_ID*** *y* ***Empleado_ID****, el conjunto de habilidades no es unívoca, sino multivaluada, ya que para cada proyecto, cada empleado puede tener más de una habilidad.*

Dependencia Funcional Compuesta

Una dependencia funcional compuesta ocurre cuando la dependencia funcional se aplica a una combinación de atributos.

Tabla de Ventas

ID_Venta	Producto	Cantidad	Precio_Total
V001	A	10	100
V001	B	5	50
V003	A	2	20

En esta tabla, la combinación de ID_Venta y Producto determina la Cantidad y el Precio_Total. Esto se expresa como:

- ***(ID_Venta, Producto) → Cantidad***
- ***(ID_Venta, Producto) → Precio_Total***

REGLAS DE DERIVACIÓN

Base de Datos I

Lic. Leandro Mariano Abraham

Clase 7 – Normalización

Las reglas de derivación en bases de datos son un conjunto de principios que permiten derivar o inferir nuevas dependencias funcionales a partir de un conjunto dado de dependencias funcionales. Estas reglas son esenciales para el proceso de normalización, que busca asegurar la consistencia y la integridad de los datos en una base de datos.

Regla de Reflexividad:

Si un conjunto de atributos Y es un subconjunto de un conjunto de atributos X, entonces X→Y

Ejemplo:

Tabla de Empleados

Empleado_ID	Nombre	Departamento	Edad
1	Ana	Finanzas	30
2	Juan	IT	25

En esta tabla, sabemos que ***{EmpleadoID,Nombre,Departamento,Edad} → Nombre*** *porque el atributo Nombre es un subconjunto del conjunto completo de atributos. (O sea, pertenece al conjunto)*

Regla de Augmentación:

Si X→Y, entonces para cualquier conjunto de atributos Z, XZ →YZ.

Ejemplo:

Tabla de Pedidos

Pedido_ID	Cliente_ID	Fecha	Monto
101	C01	1/8/2024	500
102	C02	2/8/2024	300

Si tenemos la dependencia funcional ***{PedidoID} → {ClienteID,Fecha,Monto},*** *podemos derivar que:*

{PedidoID} → {ClienteID,Fecha,Monto,Estado} *para cualquier atributo adicional como Estado.*

Regla de Transitividad:

Si X→ Y y Y→Z, entonces X→Z

Ejemplo:

Tabla de Cursos

Curso_ID	Profesor	Departamento
C01	Smith	Matemáticas
C02	Jones	Ciencias

Si sabemos que:

{CursoID}→{Profesor} *(un curso tiene un único profesor)*
{Profesor}→{Departamento} *(cada profesor pertenece a un único departamento)*

Entonces, podemos inferir que:
{CursoID}→{Departamento} *(un curso también determina el departamento).*

Regla de Unión:

Si X→Y y X→Z, entonces X→YZ

Ejemplo:

Tabla de Productos

Producto_ID	Nombre	Precio	Stock
P001	Laptop	1000	10
P002	Mouse	20	100

Si sabemos que:

{ProductoID} →{Nombre}
{ProductoID} →{Precio}

Entonces, podemos derivar que:
{ProductoID} →{Nombre,Precio}

Regla de Separación:

Si X→YZ, entonces X→ Y y X → Z

Ejemplo:

Tabla de Estudiantes

Estudiante_ID	Curso_ID	Nota	Comentario
S01	C01	85	Bueno
S02	C02	90	Excelente

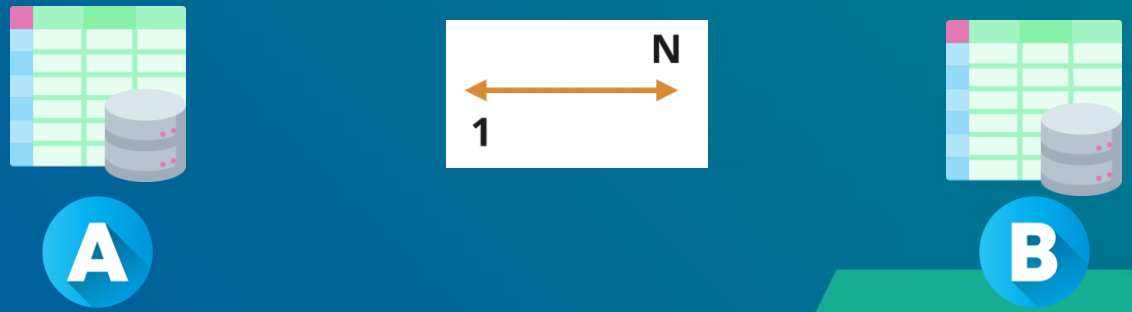
Si tenemos la dependencia ***{EstudianteID,CursoID}→{Nota,Comentario}****, podemos inferir:*

{EstudianteID,CursoID}→{Nota}
{EstudianteID,CursoID}→{Comentario}

SIGNIFICADO DE LOS ATRIBUTOS EN UNA RELACIÓN (CARDINALIDAD)

Reglas para optimizar las relaciones:

Si la entidad A contiene tuplas que se relacionan de a una con más de una tupla de la entidad B (o sea una relación de uno a muchos), la relación correspondiente a "muchos" (en este caso la entidad B) contendrá un atributo que la relacione con la otra.



Ejemplo:

Tenemos una tabla-relación que contiene datos de las provincias disponibles para residir en una convención laboral. Por otro lado, la tabla que contiene datos de los becados que concurrirán a dicha convención.

Entidad A: Provincias

Cod-Prov	Descripción
1	La Pampa
2	Corrientes
3	Neuquén
4	La Rioja

Entidad B: Becados

ID	Nombre	Cod-Prov
1	Farina, Ana	1
2	Perez, Ariel	4
3	Gómez, Luis	2
4	Valdez, María	1

Esto me indica la relación:

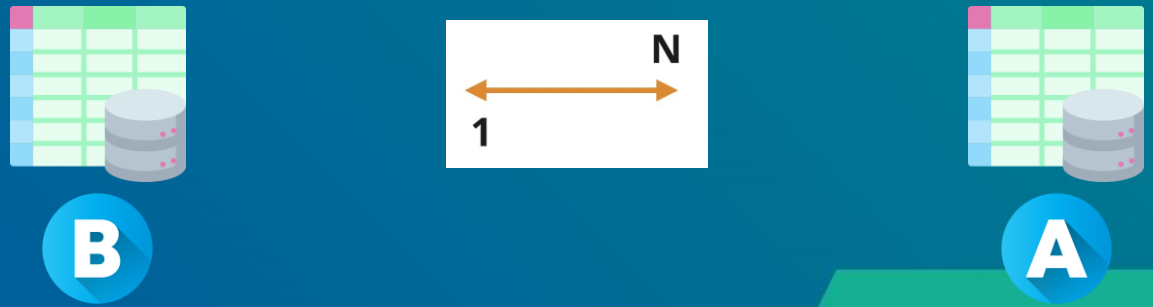
A una provincia → concurren → varios becados

Un becado → puede concurrir a → una provincia

No hizo falta la creación de una tercera tabla, ya que la vinculación se produce cuando se agrega en la entidad B el atributo COD-PROV (el cual es clave ajena referenciando a la entidad A).

Reglas para optimizar las relaciones:

Si la entidad A contiene más de una tupla que se relaciona con tuplas de la entidad B de a una (o sea una relación de muchos a uno), sucede lo mismo que en el caso anterior, en la relación de "muchos" (en este caso la entidad A) contendrá un atributo que la relacione con la otra.



Ejemplo:

Tenemos una tabla-relación que contiene datos de los empleados de una empresa. Por otro lado, tenemos la tabla que contiene los datos de las gerencias de esa empresa

Entidad A: Empleados

Legajo	Nombre	IdGerencia
1010	Torres, José	01
1012	Perez, Sergio	03
1015	Funes, Diego	02
1020	López, Alan	01

Entidad B: Gerencias

IdGerencia	Nombre
01	Sistemas
02	Contaduría
03	Finanzas
04	Recursos Humanos

Esto me indica la relación:

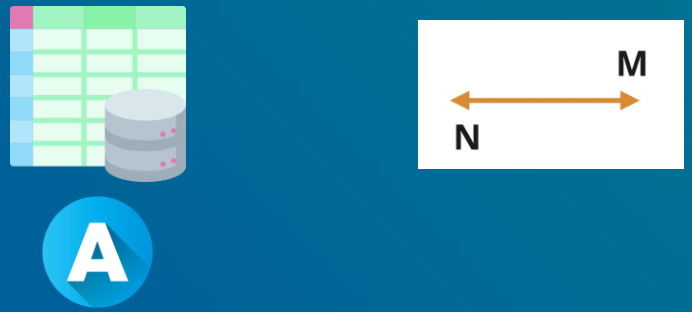
Un empleado → pertenece a → una gerencia

A una gerencia → pueden pertenecer a → varios empleados

En este caso tampoco hizo falta la creación de una tercera tabla, ya que la vinculación se produce cuando se agrega en la entidad A el atributo IDGERENCIA (el cual es clave ajena referenciando a la entidad B)

Reglas para optimizar las relaciones:

Si la entidad A contiene más de una tupla que se relaciona con más de una tupla de la entidad B (o sea una relación de muchos a muchos), se justifica la creación de una tercera tabla que interrelacione las otras 2.



Ejemplo:

Contamos con la entidad A: PACIENTES, que contiene más de una tupla que se relaciona con más de una tupla de la entidad B: MEDICAMENTOS (o sea una relación de muchos a muchos), se justifica la creación de una tercera tabla PAC-MED que interrelacione las otras 2.

Entidad A: Pacientes

Id-Pac	Nombre	Edad
21010	Torres, José	51
21012	Alonso, Sergio	46
31015	Funes, Diego	34
51020	López, Alan	18

Entidad B: Medicamentos

Id-Med	Descripción
01	Calmante
02	Colirio
03	Cicatrizante
04	Anticoagulante

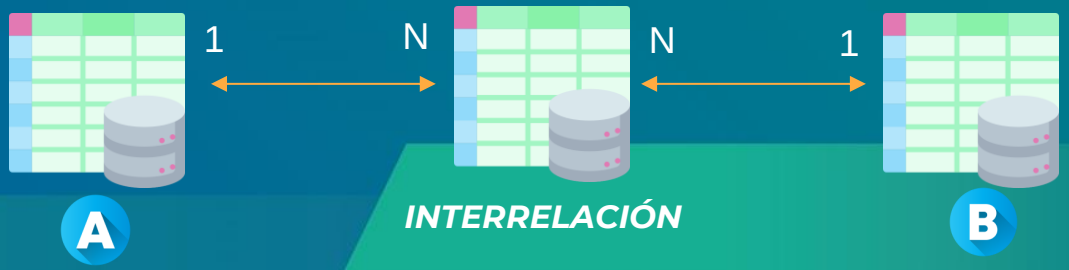
Interrelación C: PAC-MED

Id-Pac	Id-Med	Dosis-Diaria
21010	01	2
21010	03	1
21012	04	2
51020	02	3

Esto me indica la relación:

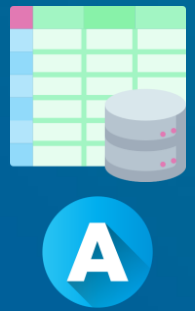
Un paciente → puede consumir → varios medicamentos

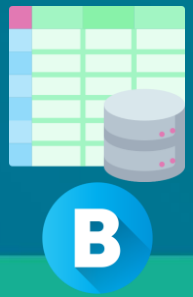
Un medicamento → puede ser consumido por varios pacientes



Relaciones de 1 a 1:

También existen relaciones de 1 a 1, esto significa que para cada fila de la entidad A, existe un único registro de la entidad B y viceversa. Un ejemplo podría ser una tabla de "personas", donde se identifican de manera univoca a cada persona a través de su N° de identidad, y una tabla de "pasaportes" donde cada pasaporte le puede pertenecer únicamente a una persona.





Valores nulos en las tuplas:

En una relación se agrupan determinada cantidad de atributos que a veces conforman una relación demasiado "grande". Si hubiera atributos que no se aplican a la mayoría de las tuplas de esta relación, aparecerán gran número de nulos. Por eso, se trata de evitar incluir en una relación, atributos cuyos valores puedan ser nulos, y de existir, que sea en su minoría.

Ej: Si sólo el 10% de los alumnos trabajan, no se justifica incluir un atributo CUIL o CUIT. Se puede crear una relación ALU-TRAB que contenga sólo tuplas con los alumnos que están empleados.

ALU-TRAB(LEGAJO, CUIT-CUIL)

FORMAS NORMALES

Edgard Codd propuso 3 formas normales, las cuales se conocen como primera (1FN), segunda (2FN) y tercera (3FN) formas normales. Todas estas formas se definen bajo las restricciones de las dependencias funcionales entre los atributos de una relación.

La normalización de los datos consiste en descomponer las relaciones distribuyendo sus atributos en relaciones más pequeñas satisfaciendo un cierto conjunto de restricciones. Es el proceso de producción de grupos óptimos de atributos en las relaciones.

Ventajas de llevar las relaciones hasta la tercera forma normal:

- *Se evitan anomalías en la inserción, borrado y modificación.*
- *Se facilita la extensión: si en un futuro se llevan a cabo ampliaciones, se tendrán menos cambios en la estructura de la base de datos.*

Primera Forma Normal (1fn)

Una relación está en 1FN si y sólo si todos sus dominios subyacentes contienen sólo valores atómicos. Significa la eliminación de grupos repetitivos.

No está en 1° Forma Normal

Legajo	Nombre	Idioma	Nivel de Idioma	Sección
120	Juan	Ingles	B	Sistemas
121	Jose	Portuges, Italiano	A,B	Contaduría
122	Ana	Ingles	C	Ventas

Legajo	Nombre	Cod-Idioma	Nombre-Idioma	Nivel-Idioma	Sección
120	Juan	01	Inglés	B	Sistemas
121	José	03	Portugués	A	Contaduría
121	José	04	Italiano	B	Contaduría
122	Ana	01	Inglés	C	Ventas

Los atributos referidos al "idioma" del empleado son los que provocan la repetición de los legajos. Entonces, eliminar los grupos repetitivos significa "llevarme a otra relación los atributos causantes de tal iteración".

Está en 1° Forma Normal

Así quedan dos relaciones entidades dispuestas de la siguiente manera:

- **EMPLEADOS (LEGAJO, NOMBRE, SECCION)** *Con los atributos propios del empleado.*
- **IDIOMAS (COD-IDIOMA, NOMBRE-IDIOMA)** *Con los atributos propios del idioma.*

Y una interrelación EMP-IDIOMA para indicar el NIVEL de cada empleado en cada idioma.

- **EMP-IDIOMAS (LEGAJO, COD-IDIOMA, NIVEL-IDIOMA)**

Tabla de Empleados

Legajo	Nombre	Sección
120	Juan	Sistemas
121	José	Contaduría
122	Ana	Ventas

Tabla Idiomas

Cod_Idioma	Idioma
1	Ingles
3	Portuges
4	Italiano

Tabla EMP - IDIOMAS

Legajo	Cod_Idioma	Nivel
120	1	B
121	3	A
121	4	B
122	1	C

Segunda Forma Normal (2fn)

Una relación está en 2FN si y sólo si está en 1FN y todos los atributos no clave tienen dependencia funcional completa con la clave primaria, no existen dependencias parciales.

Ej: Tabla Asignación de Proyectos

Está en 1° Forma Normal pero no en 2° Forma Normal.

NRO-EMP	NRO-PROY	NOMBRE-EMP	SUELDO-HORA	FECHA-INICIO
1	101	Juan	20	1/1/2024
2	102	María	30	15/2/2024
1	103	Juan	20	10/3/2024

La clave primaria es compuesta: (NRO-EMP, NRO-PROY).

- FECHA-INICIO depende de ambos atributos NRO-EMP y NRO-PROY, ya que la fecha de inicio es específica para cada proyecto.
- Pero NOMBRE-EMP y SUELDO-HORA dependen solo de NRO-EMP (el nombre y el sueldo de Juan no cambian por proyecto).

Esto significa que hay una dependencia parcial, lo que rompe la 2FN.

Segunda Forma Normal (2fn)

Debemos dividir la tabla en dos para que los atributos NOMBRE-EMP y SUELDO-HORA dependan solo de NRO-EMP.

NRO-EMP	NOMBRE-EMP	SUELDO-HORA
1	Juan	20
2	María	30

NRO-EMP	NRO-PROY	FECHA-INICIO
1	101	1/1/2024
2	102	15/2/2024
1	103	10/3/2024

Ahora está en 1° y 2° forma normal.

Tercera Forma Normal (3fn)

Una relación está en 3FN si y sólo si está en 2FN y todos los atributos no clave son mutuamente independientes entre sí.

Partimos de la tabla “Empleados” que contiene los siguientes atributos:

LEGAJO	NOMBRE	NRO-SECCION	OFIC-SECCION
1	Ana	101	A1
2	Juan	102	B2
3	Carla	101	A1

Para que una tabla esté en 3FN, primero debe cumplir con la Primera y Segunda Forma Normal:

- ***Primera Forma Normal (1FN):*** *Todos los atributos contienen valores atómicos y la tabla no tiene filas duplicadas. En este caso, la tabla EMPLEADOS cumple con 1FN.*

- ***Segunda Forma Normal (2FN):*** *La tabla debe estar en 1FN y todos los atributos no clave deben depender completamente de la clave primaria. La clave primaria aquí es LEGAJO, y los atributos NOMBRE, NRO-SECCION, y OFIC-SECCION dependen completamente de LEGAJO.*

Para que una tabla esté en 3FN, debe cumplir con 2FN y, además, ningún atributo no clave debe depender funcionalmente de otro atributo no clave.

En nuestra tabla original EMPLEADOS:

- LEGAJO → NOMBRE
- LEGAJO → NRO-SECCION
- LEGAJO → OFIC-SECCION
- NRO-SECCION → OFIC-SECCION (dependencia funcional entre atributos no clave)

La dependencia NRO-SECCION → OFIC-SECCION indica que OFIC-SECCION depende de NRO-SECCION.
Esta dependencia entre atributos no clave viola la 3FN.

Para resolver esta violación, necesitamos descomponer la tabla original en dos tablas donde estas dependencias no causen problemas:

Tabla Empleados

LEGAJO	NOMBRE	NRO-SECCION
1	Ana	101
2	Juan	102
3	Carla	101

Tabla Sección

NRO-SECCION	OFIC-SECCION
101	A1
102	B2

Verificamos la Nueva estructura:

- En la tabla EMPLEADOS, LEGAJO es la clave primaria y NOMBRE y NRO-SECCION dependen completamente de LEGAJO. *No hay dependencias entre atributos no clave.*
- En la tabla SECCIONES, NRO-SECCION es la clave primaria y OFIC-SECCION depende completamente de NRO-SECCION.

Con estas tablas, hemos eliminado la dependencia entre atributos no clave, cumpliendo así con la 3FN.

Base de Datos I

Lic. Leandro Mariano Abraham

Clase 7 – Normalización

Los números tienen una historia importante que contar. Dependen de ti, para darles una voz

- Stephen Few