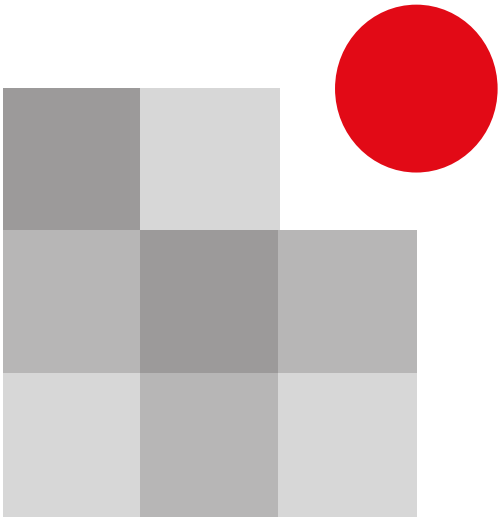
# SISTEMAS DE INFORMACIÓN II



IPLACEX
instituto profesional

## UNIDAD I

## SISTEMAS Y GESTIÓN DE BASE DE DATOS PARA LA ADMINISTRACIÓN MODERNA

# CLASE 01

## 1. SISTEMAS PARA LA ADMINISTRACIÓN MODERNA

En una empresa lo importante es el uso de capital, dinero y recursos tangibles, para generar nuevos productos. El buen empleo de la información continuará creando nuevas oportunidades de mejorar los sistemas y su buen uso ayuda a obtener ventajas competitivas, por ejemplo con nuevos productos y servicios que a la larga significa un mejor y eficaz trato a los clientes.

Para poder satisfacer las diferentes necesidades de la empresa es que existen diferentes categorías de sistemas de información.

Recordemos el concepto de "Sistemas":

"Conjunto de componentes que interactúan entre sí para lograr un objetivo común"

Los sistemas han tenido un enfoque histórico, es decir, se han preocupado de producir distintos reportes. Los más comunes son los reportes de contabilidad. Pero para saber más sobre los distintos tipos de sistemas de información, se comenzará estudiando aquellos que están orientados a un enfoque histórico y que de acuerdo a sus falencias nacieron otros más actuales. Tales sistemas se centran en los siguientes:

a) Sistemas de Contabilidad

b) Sistemas de reporte de responsabilidades

c) Sistemas integrados para procesamiento de datos

A continuación, se estudiarán cada uno de ellos.

### a) Sistemas de Contabilidad

Antiguamente las organizaciones se preocupaban solamente de los resultados y no de la información, es decir, solamente se basaban en los reportes de contabilidad y no en la información básica que controlaba las funciones de la organización.

Las empresas solucionaban los problemas utilizando los reportes de contabilidad del día anterior que les resolvía el problema en ese entonces. Con este sistema la

estructura de la empresa estaba estancada y los cambios eran en forma paulatina, sin mayor modificación.

Los sistemas de contabilidad no se preocupaban de las necesidades básicas de la empresa, es decir, de la retroalimentación de la información que permitía saber que era lo que no estaba funcionando para poder corregir los errores y poder llegar a cumplir de mejor manera los objetivos de la empresa.

Pero este error era compartido no sólo por los diseñadores sino también por los gerentes que no estaban entrenados para desarrollar una buena labor.

### b) Sistemas de Reportes de Responsabilidades

El resultado de los sistemas de contabilidad fueron los reportes realizados para la asignación de responsabilidades. Estos reportes acumulan datos históricos de contabilidad pero en intervalos de tiempo específicos de acuerdo a las actividades y niveles de responsabilidad de los departamentos de la empresa.

El reporte de responsabilidades está relacionado con las actividades que están a cargo directamente de un individuo y de las cuales él sería responsable.

Cada gerente, sin importar nivel, es responsable de cualquier pérdida y mensualmente será supervisado y evaluado por su gestión.

### c) Sistemas Integrados para Procesamiento de Datos

Al examinar los reportes de responsabilidades los diseñadores se dieron cuenta que las operaciones de los sistemas iban mucho más allá que las funciones de contabilidad que eran tan específicas y actuaban individualmente.

Se vieron en la necesidad que los sistemas tuvieran pequeños subsistemas que interrelacionaran todas las funciones de una sola vez (hombre, material, capital). Es decir, que todo estuviera relacionado para cumplir de una sola vez todas las funciones de la organización.

El resultado, el tan llamado Sistema Integrado, el cual además de tener una red de subsistemas relacionados, también cuenta con otras características como por ejemplo, una entrada única de datos para usos múltiples. Este tipo de sistema hace que los datos almacenados puedan ser usados por cualquier área funcional de la organización.

# CLASE 02

## Ejemplo Nº 1

La sección de fabricación enumera una lista de materiales, la cual puede ser utilizada para solicitar las órdenes a los proveedores, iniciar la orden de producción y fijar los precios de los productos finales

Otra de las características distintivas de los sistemas integrados de información es su facilidad de respuesta al cambio, es decir, dado que los métodos, funciones y actividades cambian constantemente según los requerimientos de la empresa, es más fácil los cambios en sistemas computarizados que en sistemas manuales. Las modificaciones pueden ser programadas, sin la necesidad de tener que entrenar al personal en cuanto a nuevos equipos y procesos.

Tabla Nº 1: Desventajas de los Sistemas de Información con Enfoque Histórico.

| TIPO DE SISTEMAS | DESVENTAJAS |
|---|---|
| Sistemas de Contabilidad | Se emplean métodos manuales para procesar datos, lo que lo hace un método muy lento. Los subsistemas se trataban individuales, y no hubo intento por integrar registros que cumplieran varias funciones. No se podían realizar análisis significativos debido a que los datos se demoraban mucho en relacionarse. |
| Sistemas de Reportes de Responsabilidad | De todas formas fue un avance con respecto al sistema contable, pero debido a que el producto resultante aún estaba orientado al reporte histórico de actividades de contabilidad no tomó la integración de subsistemas, por lo que tenía una visión demasiado estrecha todavía. |
| | Este tipo de sistemas redujo en gran |

| | |
|---|---|
| Sistemas Integrados para Procesamientos de Datos | cantidad la duplicación de datos lo que mejoró las funciones de la organización, sin embargo, la integración de las principales funciones no garantizó resultados óptimos, ya que carecía de reportes que fueran saliendo con la actualización de datos que facilitara la función gerencial. |

Estos sistemas están enfocados a la elaboración de reportes históricos, más que a la forma de reportes financieros, necesidades que no satisfacen mucho a la gerencia de una empresa. Es por esto que aparecieron nuevos sistemas de información que a continuación se detallarán.

Realice Ejercicios N° 1 al 5

Dentro de los Sistemas de Información más usados se estudiarán las siguientes:

## 1.1. Sistemas de Información Integrados

Las limitaciones que poseía este tipo de sistema con el tiempo fueron mejorados. Una de las mejoras eran las salidas de reportes para todos los niveles de la gerencia lo que producía la toma de decisiones rutinarias.

Un sistema integrado involucra mucho más que la unión de las funciones de la organización, también ayuda a la toma de decisiones del gerente y si la decisión es definida, ésta puede ser computarizada, por ejemplo, una decisión definida sería ver la necesidad de realizar un sistema ó actualizarlo para que solucione alguna necesidad de la empresa que se ha visto afectada.

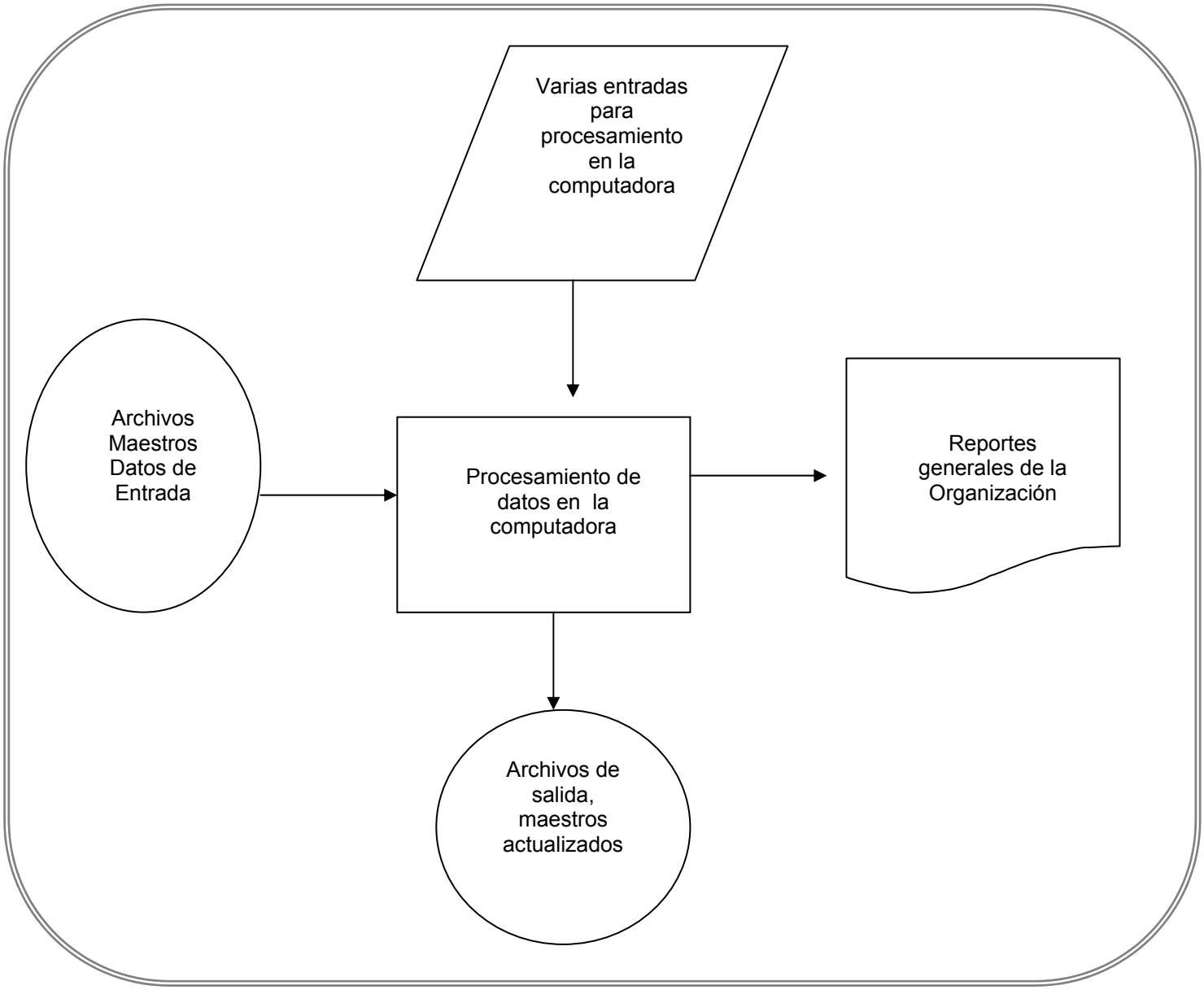
El resultado de esta mejora concretó que la gerencia planeara y controlara las decisiones en las actividades de la empresa. Todas estas actividades están basadas y se enfatizan en la planeación de las utilidades, el desempeño y el control en todos los niveles, tanto financieros como no financieros de los distintos departamentos.

Todo lo que viene siendo reportes financieros pasaron a un segundo plano, pasando a ser un subproducto de la información procesada que ayudará a controlar las

operaciones y los objetivos buscados por la organización. El procesamiento de datos tendrá múltiples usos, lo que va reducir los costos.

La característica principal de los sistemas de información integrados incluye la producción de una salida con significado para la gerencia, que ayude a la toma de decisiones rutinarias. Las transacciones que son comunes en todos los departamentos se almacenan en un archivo de datos común. Aquellos datos que afectan a más de un área funcional se capturan solo una vez y después son procesados para todos los usuarios que requieran la información.

Ejemplo Nº 2



En el ejemplo se muestra un Sistema Integrado en dónde los datos son almacenados en archivos maestros, los cuales corresponde a archivos de sistema de base de datos que contienen toda la información de la organización, los datos que se pueden ingresar son archivos de ingeniería, archivos de cuentas por pagar, nómina de empleados, inventarios y precios, investigaciones y sus desarrollos. Éstos se procesan en la computadora para generar reportes tales como ventas, balances, la rentabilidad de los productos, desempeño del producto, situación y faltante de materias primas, entre otros.

## 1.2. Sistemas de Información en Tiempo Real

Son sistemas también denominados como de Tiempo Real en Línea. Las aplicaciones típicas de estos sistemas son cuentas por cobrar, sistemas de reservación de pasajes en las líneas aéreas, la contabilidad de los depósitos y retiros bancarios, registros de pacientes de hospitales.

Una característica esencial es que toda la información se encuentra en línea, ésta se procesa en una computadora a penas se encuentra elaborada, de manera que se pueda acceder a ella y si se produce un cambio se pueda realizar en el instante.

El personal de la empresa al momento de requerir alguna información, recibirá de forma inmediata la respuesta para satisfacer el requerimiento. El tiempo de respuesta puede variar de segundos, horas o hasta días dependiendo de cada situación.

Es un sistema que mantendrá la información al día, permite el intercambio de información con la base de datos y además permite al programador interactuar con ella si es que ha ocurrido alguna transformación, de manera de mantener la información fidedigna y ayudar a estar alerta a los supervisores sobre ciertas áreas críticas que requieran de atención inmediata.

Posee algunas características notables que a continuación se explicarán:

a) Integración de Subsistemas:

Un sistema en tiempo real debe estar integrado para su buen funcionamiento, es decir, los datos adquiridos de una fuente pueden estar presentes en muchos subsistemas, puesto que si no se integran existirían muchos movimientos innecesarios que llevarían a un gran costo, sobre todo de tiempo.

Cada subsistema se relaciona con sus propios datos sin aprovechar las ventajas del procesamiento derivado de otros subsistemas.

# CLASE 03

b) Elementos comunes de datos:

Los datos que se acumulan debido a las distintas transacciones detalladas en línea, se conocen comúnmente como Base de datos. Además de tener todos los datos recopilados en un solo lugar, los mismos datos son útiles para muchos departamentos.

## Ejemplo Nº 3

La base de datos de un inventario puede ser usado por los departamentos de ventas, de control de la producción, de compras y finanzas.

c) Reportes a la gerencia baja y media:

Una base de datos puede ser manejada por los distintos niveles gerenciales. Los sistemas satisfacen todos los niveles, en cuanto a las actividades de la organización que tengan que ver con sus objetivos. Es por eso que los sistemas de tiempo real se enfocan en áreas como por ejemplo, cómo mejorar servicios al cliente, optimizar el presupuesto de la empresa, cómo optimizar la utilización de las instalaciones de la producción, desempeño de proveedores y mejorar las negociaciones con los sindicatos.

d) Dispositivos de Entrada y Salida en Línea:

Estos dispositivos están ubicados en las operaciones de la organización. Estos son capaces tanto de recibir como de enviar información, pueden estar muy alejados unos de otros, pero mediante una red de comunicaciones de datos que une una computadora local con otra regional y éstas con la computadora central hace que se obtenga la información al instante, en tiempo real.

Este sistema de tiempo real al ser un sistema de control anticipado mantiene los elementos disponibles de la base de datos en línea, de manera que se puedan utilizar cuando se requieran.

Las bases de datos se actualizan de manera inmediata a medida que van ocurriendo los eventos en la organización y puede ser interrogada por cualquier terminal de entrada o salida.

Esto permite que los distintos departamentos trabajen con la misma información, lo que permite una mayor coordinación con respecto a las funciones dentro de la organización.

## 1.3. Sistemas Distribuidos para el Procesamiento de Datos.

Una de las tendencias más importantes en la actualidad es el procesamiento distribuido de datos empleados por las organizaciones.

El procesamiento distribuido puede definirse como la capacidad de cómputo de bajo costo, a través de los distintos puntos de entrada de datos con distintos sistemas computacionales a través de una red de comunicaciones.

Este tipo de sistema es requerido para aquellos departamentos de la organización donde se requiera operaciones de procesamiento de datos en forma eficiente y económica.

También es una alternativa para el procesamiento centralizado de datos, debido al bajo costo de los microprocesadores, las microcomputadoras y los computadores personales.

Dentro de lo que es la historia de las computadoras la mayoría de ellas dependían de unidades centrales de procesamientos, que se programaban para realizar muchas funciones de acuerdo se necesitaban para los requerimientos. Esto provocó un serio problema en las unidades de entrada, lo que producía que la retroalimentación de la información para manejar alguna necesidad inmediata ocurriera después de grandes retrasos.

Con este problema nace la Regla 80-20, que nace de la idea que se contara con una capacidad de cómputo que se dedicara a los niveles inferiores de la organización. Esto era llevar a cabo el 80 % del trabajo y de los resultados.

Esto sucedería siempre y cuando el procesamiento distribuido de datos tenga un sentido económico de un 80 % de los datos generados. De lo contrario, otros métodos de procesamiento de datos serán los más adecuados para satisfacer los requerimientos de la organización.

Una característica de los sistemas distribuidos de procesamientos de datos es la Red distribuida de Procesamiento. Una característica de esta red es el uso de un equipamiento para la transmisión de datos desde procesadores remotos hasta las

instalaciones centrales de cómputo. Para poder operar de una buena manera el sistema deberá:

- Incorporar un cierto número de computadoras que se combinarán para formar un centro de procesamiento de datos en red. Una red de este tipo significa una gran herramienta de procesamiento de datos tanto local como regional.

- Tener una entrada en línea de datos, para proveer de la información necesaria para trabajar con múltiples funciones. Un centro operativo cumple con una gran función en mantener comunicación con el centro operativo y distintos centros de procesamientos para resolver los requerimientos que se necesiten en el instante, a través de una red de comunicación.

- Permitir el desarrollo de aplicaciones fáciles para un personal de apoyo en dónde puedan resolver problemas con la simplicidad de las operaciones.

- Permitir la generación de reportes dentro de la red que permita a la gerencia estar al tanto de las operaciones, y su buen funcionamiento.

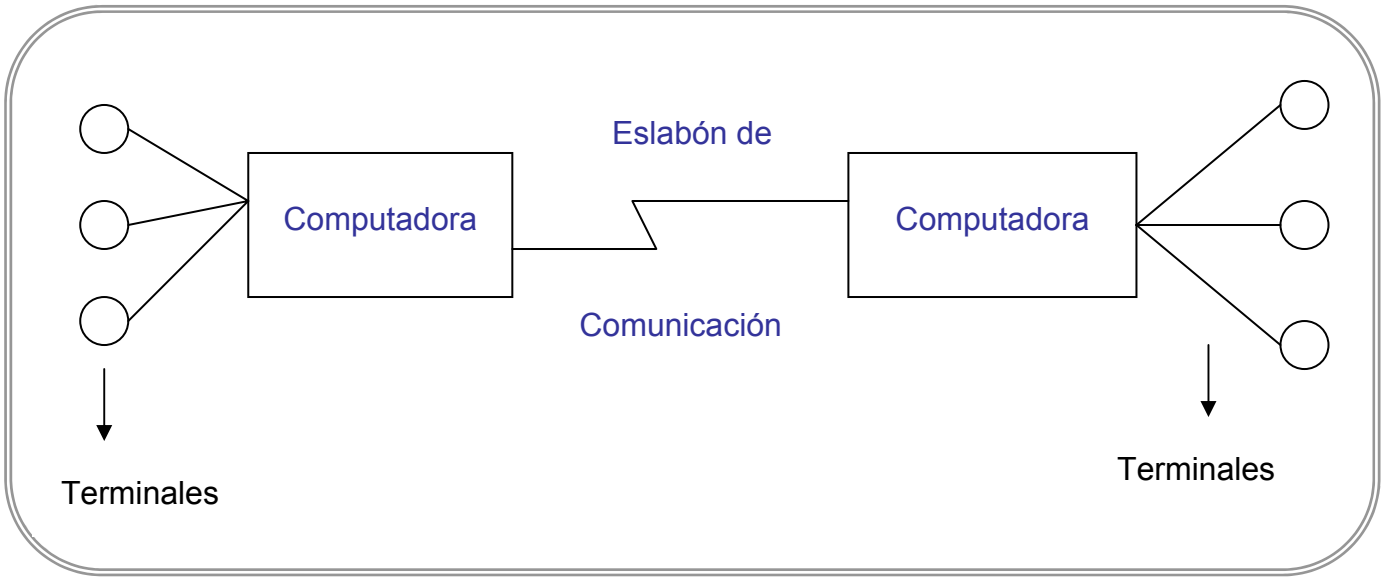
Basados en estas a características se pueden desarrollar distintas configuraciones de redes para cumplir con la transmisión de datos. Los tipos de redes más comunes son:

## CLASE 04

### a) Red Punto a Punto:

Es una de las redes más simples, existe sólo una línea de comunicación para unir dos computadoras, como se muestra en la siguiente figura.

Figura N° 1:

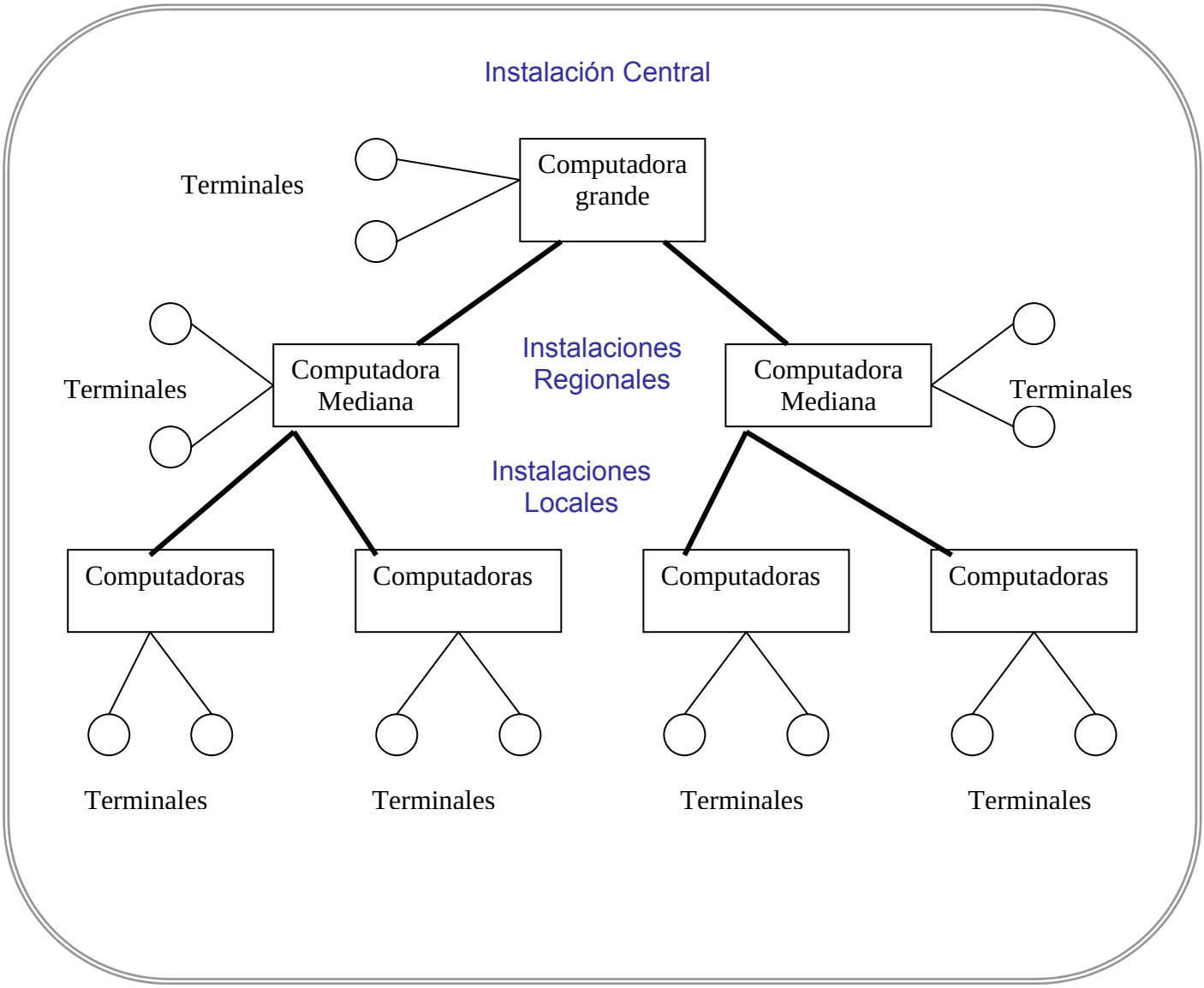


En la figura se observa el tipo de sistemas distribuidos de procesamiento de datos de Red punto a punto, en donde una computadora puede ser usada para procesar la información y la otra para llevar el control de las comunicaciones. Transmitiendo la información necesaria a los distintos terminales que necesiten de algún requerimiento.

b) Red Jerárquica o de Árbol:

En una red jerárquica los computadores realizan una función determinada y están unidos con otra computadora, generalmente de mayor tamaño. Esta computadora monitorea las actividades de las demás computadoras que se encuentran en instalaciones regionales.

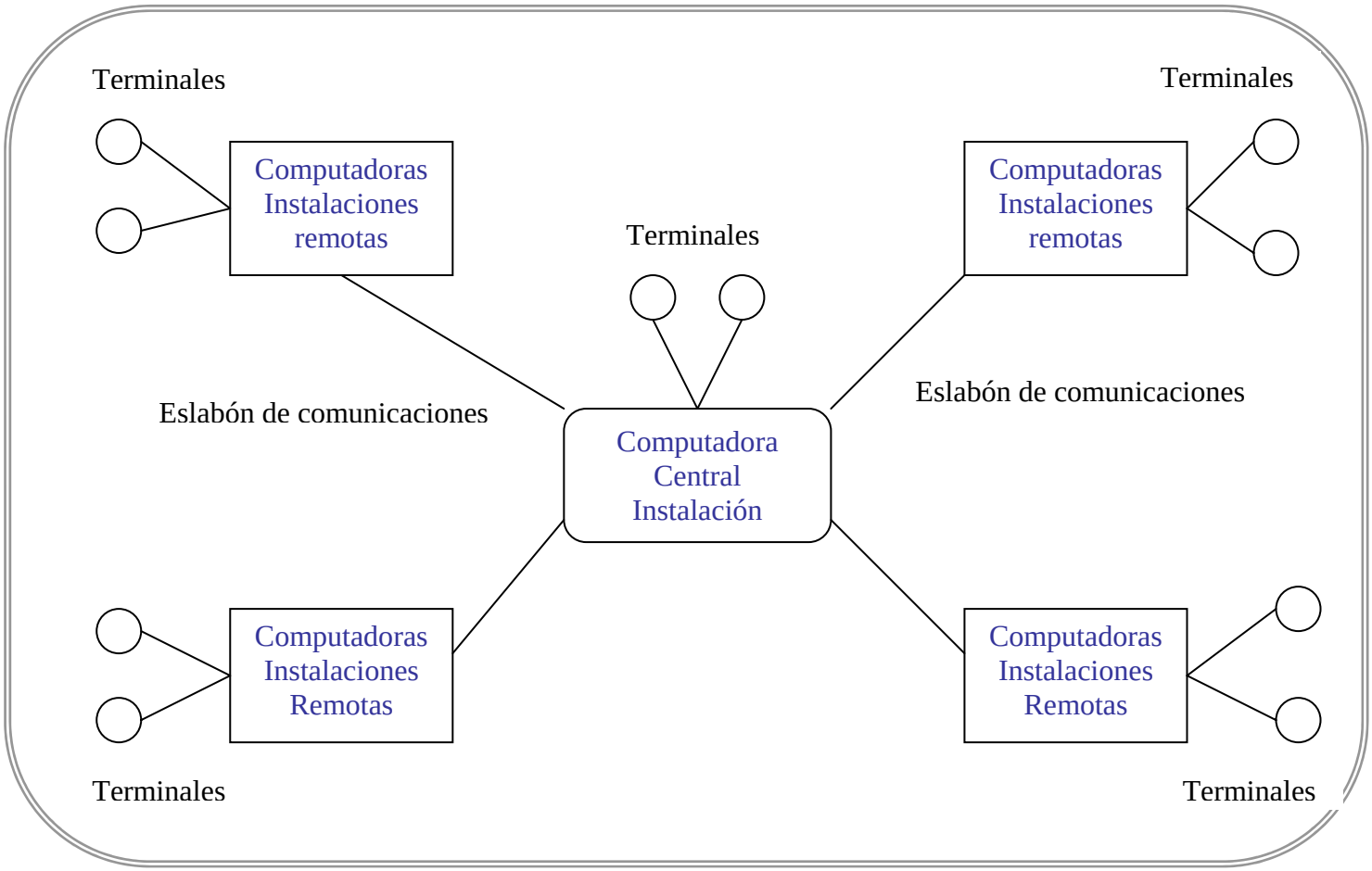
Figura N° 2:



c) Red de Estrella :

En este tipo de configuración de redes todos los computadores se reportan con las instalaciones centrales. Desplaza la información sólo en dos direcciones de ida y regreso entre los computadores remotos y los centrales.

Figura N° 3:



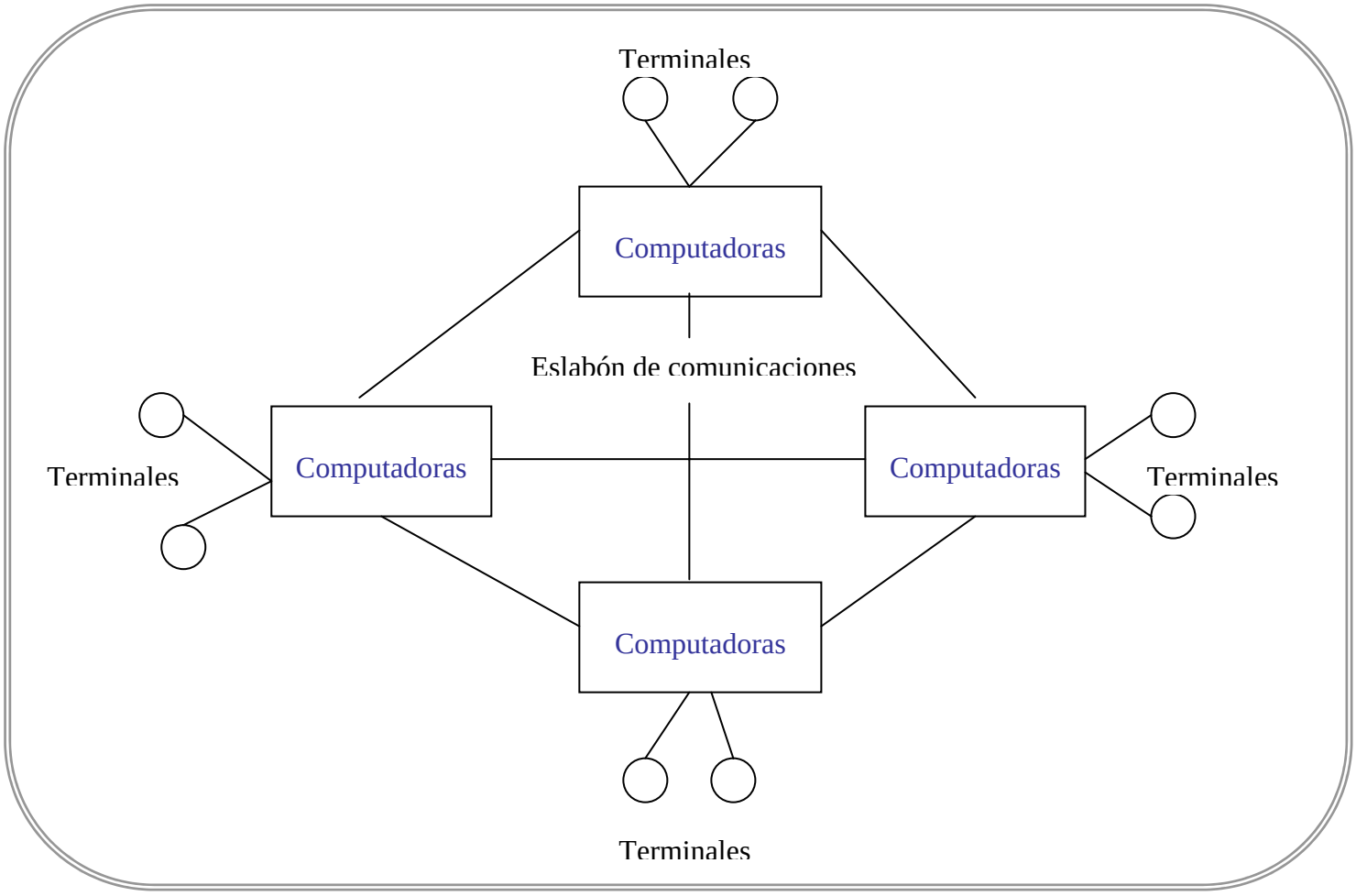

En la figura N° 3, se presenta una Red en estrella, en donde existe una computadora grande, conectada a unas más pequeñas, en donde hay un intercambio de información para satisfacer necesidades. Si alguna de las computadoras pequeñas necesita alguna información del archivo maestro consulta al computador central y obtiene los datos.

# CLASE 05

## d) Red de Circuito Cerrado :

Para este tipo de Red los computadores cuentan con una gran capacidad de procesamiento de datos, tanto local como regional. Se comunican directamente unos con otros interactuando a través de la información entre ellos.

Figura N° 4:




La figura de Red de circuito cerrado, se puede aplicar a los Bancos, debido a que si hay algún computador que no funciona, cualquier otro terminal sirve para procesar la información de algún cliente que se requiera al instante, hasta que se reestablezca el equipo con problemas.

e) Red de tipo Híbrido :

Con una mezcla de las redes mencionadas anteriormente, se puede obtener un tipo de red híbrido. Que permite que a través de una configuración de red estrella se acceda a una estrecha relación entre computadores de distintos locales y se pueda intercambiar información y mantener actualizados los datos para los distintos terminales.

Algunas otras características de Sistemas distribuidos de procesamiento de datos es establecer bases distribuidas de datos, en donde, los archivos del usuario son colocadas en donde ocurran las transacciones. Por lo tanto, siempre estará la información disponible. Este tipo de base de datos permite que los datos se encuentren en donde se necesiten dentro de la organización.

El fin del procesamiento distribuido de datos es que ayude a que los sistemas de información computacionales resulten más amenos y tengan una mayor facilidad de respuestas para las personas. En lugar de tener una o dos centrales de procesamientos que pueden resultar poco amigables, añade varias computadoras unidas entre sí en donde los terminales reaccionarán con mayor rapidez. Información que si se requiere al instante con esta medida se obtiene sin problemas de acuerdo a los datos de entrada que se han requerido.

## 1.4. Sistemas de Apoyo para la Toma de Decisiones

Con el propósito de satisfacer las necesidades de información de la gerencia que tenga en el futuro, nace el sistema de información para la toma de decisiones, fue pensado principalmente para apoyar la toma de decisión de la parte gerencia y para el personal que rediseña los sistemas de estructuras de trabajo.

Muchas veces se apoyan en sistemas de información anteriores y utilizan algunos modelos matemáticos orientados a la ciencia administrativa.

En este tipo de sistemas es necesario tener un principio "Administración por percepción". Este principio se refiere a la percepción que debe tener la gerencia para determinar tendencias futuras, tanto internas como externas, saber el impacto que causarán y poder preparar un buen desempeño en la organización.

También existe la "Administración por excepción", que determina la comparación de los resultados después de los hechos.

Con el principio de la percepción, la gerencia puede ajustar sus planes estratégicos para nuevos productos y realizar planes tácticos para estos productos.

Algunos ejemplos utilizados por el principio de administración por percepción serían:

- Establecer objetivos a largo plazo para la organización y que estén bien integrados en los distintos subsistemas de la organización.
- La evaluación de los nuevos productos a través de sus ciclos de vida.
- Evaluación del personal de la organización para que sea la adecuada para realizar el trabajo y cumplir los objetivos de manera apropiada.
- La distribución de la capacidad de la planta productiva de manera que sea la más conveniente para la organización.
- Determinar necesidades de materiales para reducir los costos de la organización.
- Adquirir fondos de capital según sean requeridos por la organización.
- Indicar mejoras de ingresos y costos operativos de la organización.

Otra de las características importantes de los sistemas de apoyo de decisiones es la relación hombre-máquina, es decir, que la persona encargada de las decisiones a tomar debe estar atento a todos los procesos de solución de los problemas que se presenten en la organización y le permite además darse cuenta de problemas que antes no eran obvios.

De esta manera la persona encargada de la toma de decisiones interactúa con el computador de manera que no sea la computadora quien tome la ultima decisión, más bien se le permite a la persona que pueda obtener varias respuestas que le permite tener varias alternativas.

Realice Ejercicios N° 6 al 16

# CLASE 06

## 2. LENGUAJE DE DEFINICIÓN Y MANIPULACIÓN DE DATOS.

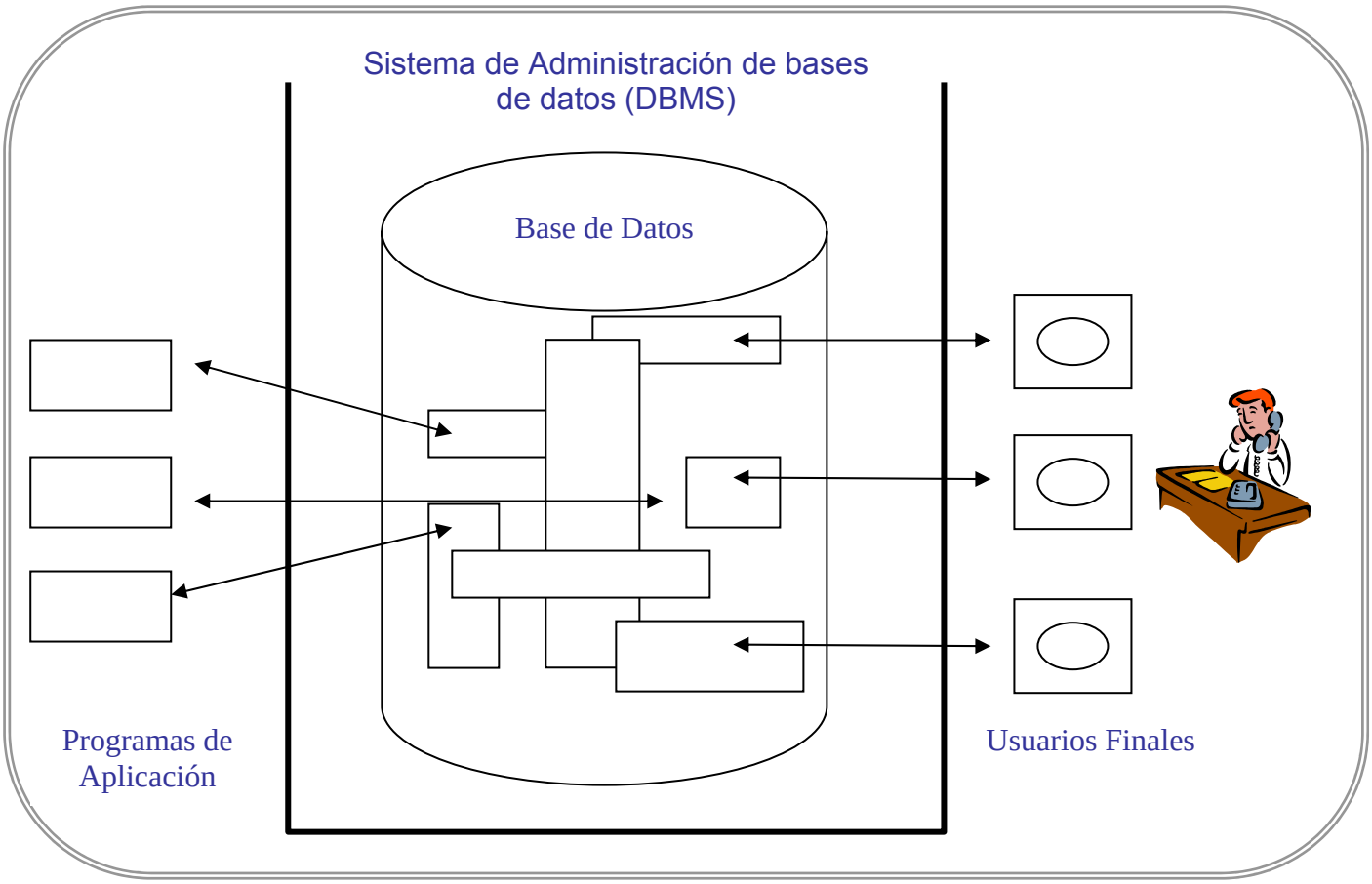
Para comenzar con este punto se explicará qué es un Sistema de Base de Datos. Éste es una de las áreas de la computación, que más se ha desarrollado. En esencia, una base de datos es un sistema de mantención de registros y de información.

Esta información puede estar relacionada con cualquier cosa que sea significativa para la organización en dónde opera el sistema. La base de datos será cualquier información que sirva para los procesos de toma de decisiones.

Con una base de datos nos es mucho más fácil compartir los datos con las distintas aplicaciones. Los analistas además de preocuparse del diseño de los archivos, establecer contenidos y fijar los métodos apropiados para la organización de datos, también deben de diseñar los medios de interacción con las bases de datos de la empresa.

En la siguiente figura se mostrará como funciona una base de datos.

Figura N° 5:



De la figura presentada se puede deducir que tanto los programas de aplicación como los usuarios utilizan la información almacenada en la base de datos para resolver algún problema con información que requieran al instante.

La parte donde reside la base de datos es en el hardware (disco duro) que posee un gran número de volumen de almacenamiento. La parte que maneja todas las solicitudes de acceso a la base de datos, recibe el nombre de Sistema de Administración de Base de Datos ó DBMS.

La característica importante de utilizar una base de datos es que proporciona a la empresa un sistema centralizado de los datos con que operará. A cargo de este sistema debe de estar una persona que controle los datos de la operación, esta persona recibe el nombre de Administrador de datos o DBA, la cual tiene una gran responsabilidad, ya que debe de tener una destreza técnica y ser capaz de entender los requerimientos de la gerencia y poder cumplir con los objetivos de la organización.

Algunas ventajas de tener un control centralizado o mejor dicho una Base de Datos son las que a continuación se explicarán:

- Reducción de Redundancia:

Si en un sistema no se usara Base de Datos se originaría una redundancia de datos debido a que cada aplicación tendría sus propios archivos.

Ejemplo Nº 4

Una aplicación del personal y otra de registros de los administrativos, cada uno poseerá un archivo que contenga la información del departamento de empleados. Ahora esto es innecesario puesto que estos archivos pueden integrarse formando un solo archivo y así eliminar redundancia. El DBA debe tener el control global de la situación y estar conciente de los requerimientos de información impidiendo la redundancia.

- Puede evitarse la inconsistencia:

Con el ejemplo anterior, suponga que los datos de un empleado tiene dos entradas distintas en la base de datos y el sistema no está al tanto de esa duplicación, es decir, no se ha controlado la redundancia, por lo tanto al querer actualizar la información del empleado sólo lo hará para una entrada; es aquí cuando se dice que la Base de datos es inconsistente y esto puede ocasionar que se entregue información incorrecta.

Ahora si el sistema le avisa de la actualización de datos para ambas entradas, entonces aún con redundancia pero controlada no habrá inconsistencia.

- Los datos pueden compartirse:

Todas las aplicaciones pueden trabajar con la misma Base de datos, no es necesario crear otras nuevas para que se ocupe la misma información.

- Pueden hacerse cumplir las normas establecidas:

Es función del DBA garantizar que se cumplan normas establecidas para representar los datos, ya sea normas de la compañía, departamentales, nacionales, etc. Es algo factible para el intercambio de datos entre sistemas.

- Aplicación de restricciones de seguridad:

Al tener jurisdicción completa sobre los datos, el DBA puede asegurar el único medio de acceso a la Base de datos y define controles de autorización para que se apliquen cada vez que se requiera acceder a la Base de datos.

- Puede conservar la Integridad:

La inconsistencia es una falta de integridad en la Base de datos, ésta debe tener información garantizada, que no tenga redundancia, pero aún cuando ésta elimine la base de datos puede contener datos incorrectos. Es entonces que el control centralizado de la Base de datos ayuda a evitar estas situaciones y el DBA deberá, mediante procesos de validación, ejecutarlos cada vez que haya alguna actualización de datos.

- Pueden equilibrarse los requerimientos contradictorios:

Cuando el DBA conoce los requerimientos globales de la empresa, independiente de los requerimientos de los usuarios, puede estructurar los servicios de la base de datos para brindar una mejor asistencia a los requerimientos de la empresa. En resumen, puede elegir aplicaciones que ofrezcan un rápido acceso.

Otra característica importante dentro de lo que es la manipulación de datos es la Independencia de datos.

Para entender con mayor facilidad la Independencia de datos se analizará lo opuesto, es decir, la dependencia quiere decir cómo los datos dependen de la manera cómo se organizan en la Base de datos del sistema de aplicación.

Ejemplo Nº 5

Puede ser que una información se almacene en forma secuencial en la base de datos y para poder acceder a ella, se requiere de un índice secuencial y la consulta se hará de acuerdo a este conocimiento. No hay otra manera de acceder a la información, es por eso que esta situación sería una dependencia de datos por la estructura de almacenamiento.

La independencia de datos es un objetivo esencial en los sistemas de las bases de datos, ya que vendría siendo la inmunidad a los cambios en la estructura de almacenamiento, lo que indica que las aplicaciones no dependen de ninguna estructura de almacenamiento, se permiten cambios para acceder a la información de distinta manera sin alterar los datos.

# CLASE 07

## 2.1. Arquitectura de un Sistema de Bases de Datos

La arquitectura de una Base de Datos se divide en tres niveles, los cuales son:

a) Interno: es el que concierne a cómo se almacenan físicamente los datos.

b) Externo: es el que atañe a cómo ve cada usuario el almacenamiento de los datos.

c) Conceptual: es el que iguala a los otros dos niveles. Si el nivel externo se relaciona con las vistas de los usuarios individuales, el conceptual ve las vistas de la comunidad de los usuarios. Y habrá sólo una vista interna que tiene que ver con toda la base de datos, tal cual está almacenada.

Un usuario se interesará sólo en una parte de la base de datos total, esta parte siempre será abstracta, si se piensa en cómo está almacenada físicamente la base de datos.

La vista de un usuario individual recibe el nombre de vista externa.

Ejemplo Nº 6

Un usuario puede ver la base de datos como registros con características del departamento, más registros con características del empleado y desconocer otros registros como por ejemplo de los proveedores y partes que ven otros usuarios de otros departamentos. Entonces, la vista externa serán las características de múltiples tipos de registros externos.

La vista conceptual, es una representación total de la base de datos, como se menciona anteriormente. Lo que se pretende es que sea una vista en la que se vean los datos como realmente son, y no como los ven los usuarios dependiendo del lenguaje (cobol) o el hardware que utilizan.

La vista conceptual se compone de múltiples características de los registros tanto de empleados como proveedores, de los distintos departamentos, de toda la base de datos de la organización.

La vista conceptual se define por medio del esquema conceptual, el cual incluye las definiciones de todos los registros conceptuales. Por lo tanto, la vista conceptual es una vista del contenido total de la base de datos y el esquema conceptual es la definición de esa vista también utilizando un lenguaje el DDL conceptual ó lenguaje de definición de datos conceptual.

El nivel interno de la arquitectura, es el nivel más bajo de la base de datos, se componen de características de registros internos o de almacenamiento. Se mantiene a un paso del nivel físico y supone en esencia un espacio direccional infinito.

La vista interna se describe por medio del esquema interno, el cual no sólo define los diversos tipos de registros almacenados, si no que además define los índices existentes, en qué secuencia se hayan los registros, qué campos hay almacenados, etc. Se utiliza usando otro lenguaje de definición de datos: DDL interno.

Siguiendo en la arquitectura de un sistema de base de datos, se analizará el Sistema de administración de datos y el Administrador de datos.

a) Sistema de administración de datos:

Es el software que maneja todos los accesos a la base de datos.

Ejemplo Nº 7

Un usuario solicita un acceso a la base de datos mediante un lenguaje de manipulación de datos, el DBMS detiene la solicitud y la interpreta, además inspecciona por turno el esquema externo, luego el esquema conceptual y posteriormente el esquema interno, con esto define la estructura de almacenamiento. Finalmente realiza las operaciones pertinentes en la base de datos.

b) El administrador de datos:

Es la persona o grupo de personas encargada del control general del sistema de base de datos. Entre sus funciones se destacan las siguientes:

1) Decidir el contenido de la información de la base de datos:

Decidir que información se mantendrá en la base de datos, identificar las entidades de interés para la empresa y la información que debe registrarse de las entidades. Después de hacer esto el DBA debe definir la base de datos escribiendo el esquema conceptual mediante el lenguaje de manipulación de datos conceptual.

2) Decidir la estructura de almacenamiento y la estrategia de acceso:

Mediante el lenguaje de definición de datos interno debe decidir la estructura de almacenamiento de la base de datos. Especificar la asociación entre la definición de la estructura y el esquema conceptual.

3) Vincularse con los usuarios:

Es responsabilidad del DBA vincularse con los usuarios y mantener la información actualizada para ser utilizada por ellos, todo mediante el lenguaje de definición de datos externo.

4) Definir una estrategia de respaldo y recuperación:

Cuando una empresa adopta un sistema de base de datos depende mucho de él. En el caso que alguna parte no funcione se deben tomar medidas rápidas para reparar los daños y reducir al mínimo los problemas para todo el sistema.

5) Controlar el desempeño y responder a los cambios de requerimientos.

Debe organizar de tal manera el sistema para que sea lo mejor para la empresa. También realizar cualquier ajuste si los requerimientos cambian.

El DBA necesitará varios programas de rutina para realizar estas tareas, como por ejemplo:

- Rutinas de carga, que crea la versión inicial de la base de datos.
- Rutinas de reorganización, reorganizar la base de datos y recuperar espacios que han sido ocupados por datos antiguos.
- Rutinas de registro de eventos, anotar cada operación que se realiza en la base de datos en conjunto con el usuario que realizó la operación.
- Rutinas de recuperación, arreglar la base de datos después de algún problema de hardware o programa.
- Rutinas de análisis estadístico, para controlar el desempeño.

# CLASE 08

## 2.2. Diseño en un Ambiente de Base de Datos

Aunque los analistas no diseñan las bases de datos, a menudo realizan sistemas que implican los DBMS y bases de datos.

En la mayoría de las empresas donde se manejan grandes bases de datos y se utilizan bien, el personal responsable de trabajar con la base de datos vigila el diseño y desarrollo, define su esquema, mantiene el diccionario de datos y busca esquemas para los datos.

El analista de sistemas debe además determinar los requerimientos necesarios de la información y determinaciones de los procedimientos de los programas. Se analiza un subesquema que es la definición lógica de los datos para la base de datos que utilizará el programa. Estos datos constan de nombres y descripciones y es un subconjunto del esquema.

Para cada base de datos existe un único esquema. Pero pueden existir muchos subesquemas. Cada aplicación de sistemas de información que utilice la base de datos puede tener un subesquema diferente.

Ahora el analista de sistemas es el encargado de determinar cómo se vincularán los datos para cumplir con los requerimientos de los usuarios.

El analista debe de analizar mucho los requerimientos, prever consultas complejas y ver cómo poder proporcionar métodos para solucionarlos.

Las estructuras de los registros son determinados por el analista de sistemas. Él debe saber qué datos son los necesarios para los usuarios, los cuales les puedan proporcionar una información deseable.

Sin embargo, como en una base de datos las estructuras de registros son lógicas (conociendo la forma en que el programa verá los datos) pueden ser distintas a la estructura física, más real.

El DBMS ayudará a realizar la transformación necesaria de los datos para dar al programa lo que espera. Si un sistema utiliza una base de datos existente, el analista debe seguir los siguientes pasos:

- Familiarizarse con el esquema de la base de datos.
- Revisar las especificaciones en el diccionario de datos.
- Determinar los requerimientos lógicos de los datos y en conjunto el personal de la administración de datos, desarrollar el subesquema que se adecue al esquema que proporcionan los datos requeridos por el sistema de información.
- Identificar y crear los archivos maestros necesarios para el sistema de información, pero que no forman parte de la base de datos existente.
- Determinar los procedimientos adecuados de procesamiento para todas las entidades, ya sea que ellas estén en una base de datos o archivos separados.
- Considerar en el diseño preocupación por los métodos de entrada y salida.

Si los analistas están desarrollando una nueva base de datos todos los pasos anteriores son necesarios, puesto que va a necesitar un acercamiento con el personal que trabaja con la base de datos, así como también ver definiciones y diccionarios de datos.

El DBMS no reemplaza los principios de diseño de sistemas, más bien proporciona al analista una herramienta adicional para usarla en el desarrollo de sistemas.

Las preguntas acerca de los requerimientos de los usuarios ayudan mucho a tomar cualquier decisión, con respecto al desarrollo del sistema o el uso de algún software de base de datos.

Así el uso del DBMS no es un objetivo, sino un medio para lograr un fin.

### 2.3. Bases de Datos Distribuidas

Es una base de datos que no está totalmente almacenada en un mismo lugar físico, sino que se distribuye a lo largo de un conjunto de computadores geográficamente separados y éstos mediante redes de comunicaciones se conectan.

Ejemplo Nº 8

Un sistema de base de datos de alguna tienda comercial donde los datos de los clientes se distribuye por todas las sucursales, pues facilita al cliente comprar en cualquier ciudad, y permite al vendedor acceder a la información correspondiente para comprobar los datos del cliente.

Las ventajas que ofrecen un sistema centralizado es que comparte los datos, sin embargo su gran desventaja son sus costos muy elevados, los cuales se requieren para implementar una unidad de comunicación así.

Su objetivo básico es que se perciba que es un sistema centralizado, que los datos son independientes en la manera como se distribuyen, es decir, posee independencia de datos.

## CLASE 09

### 3. LENGUAJE SQL

La operación fundamental de SQL es la transformación, que se representa con las sintaxis de bloques como SELECT-FROM-WHERE. (Según Date).

Por ejemplo:

Obtener números de proveedores de la tabla S# de la base de datos de una empresa, donde el total de proveedores es de 10 personas que vienen de distintas ciudades entre ellas Santiago, además obtener su estado para la ciudad de Santiago. La sintaxis[1] para este ejercicio sería el siguiente:

```
SELECT    S #,  ESTADO
FROM      S
WHERE     CIUDAD = "Santiago"
```

[1] Sintaxis se le denomina a la operación que se le pide a la base de datos que ejecute en la búsqueda de información.

S se le denomina a la tabla de la cual se extrae la información.

Se obtiene el siguiente resultado:

TABLA FISICA

| S# | ESTADO | CIUDAD |
|---|---|---|
| S1 | 5 | TALCA |
| S2 | 10 | SANTIAGO |
| S3 | 30 | SANTIAGO |
| S4 | 20 | TALCA |
| S5 | 40 | RANCAGUA |
| S6 | 18 | RENGO |
| S7 | 20 | RENGO |
| S8 | 55 | TALCA |
| S9 | 47 | RANCAGUA |
| S10 | 13 | TALCA |

| S # | ESTADO |
|---|---|
| S2 | 10 |
| S3 | 30 |

En el ejemplo lo que se pide es que se busque en la base de datos todas las reglones en donde CIUDAD = "SANTIAGO" y que de esa condición se extraiga el número del proveedor (S #) y su ESTADO.

Ahora se examinarán las principales características del lenguaje de recuperación, SQL con varios ejemplos bien sencillos para un mejor entendimiento.

a) Recuperación Simple: se obtienen los números de todas las partes suministradas.

```
SELECT  UNIQUE P#
FROM    S P
```

Se obtiene:

| P# |
|---|
| P1 |
| P2 |
| P3 |
| P4 |
| P5 |
| P6 |

Lo que se pide en el ejemplo anterior es que desplieguen todos los números de partes (P#), que existan y para que no hayan duplicados en SQL se utiliza la palabra clave UNIQUE. Por lo tanto en la lista no habrá dos veces P2 o cualquier P#.

b) Otro ejemplo de recuperación Simple: es obtener todos los detalles de los empleados.

Pueden ser de dos maneras:

```
SELECT    *
FROM      S
```

o también:

```
SELECT    S #, NOMBRE, DIRECCIÓN, CIUDAD
FROM      S
```

El asterisco es una abreviatura de incluir todos los datos ordenados de todos los campos, o bien, puede uno especificar los campos a desplegar. De las dos maneras está correcto.

c) Recuperación Calificada: es aquella de la cual se obtienen datos específicos, por ejemplo los números de los empleados para los que vivan en Talca y que tengan una edad mayor a 20 años.

```
SELECT    S #
FROM      S
WHERE     CIUDAD = "TALCA"
AND       EDAD  >  20
```

Se obtiene:

| S # |
|---|
| S3<br>S4 |

La condición WHERE puede incluir operadores como = , >, <, (no igual), >=, <=; y los operadores boléanos AND, OR y NOT. Para llevar un orden de las evaluaciones se utiliza paréntesis. En este ejemplo sólo se encontraron dos empleados que cumplían con las condiciones señaladas.

d) Recuperación con ordenamiento: por ejemplo obtener el número del empleado y estado de los que vivan en Talca, en orden descendente.

```
SELECT    S #, ESTADO
FROM      S
WHERE     CIUDAD = "TALCA"
ORDER     BY  ESTADO  DESC
```

Se obtiene:

| S # | ESTADO |
|---|---|
| S3 | 30 |
| S2 | 10 |

Para poder dar un orden a la secuencia el usuario debe de especificarlo antes de que sea desplegado. DESC, quiere decir en forma descendente, ASC quiere decir, ascendente.

e) Recuperación de más de una tabla: por ejemplo para cada proveedor obtener su número y los nombres de todas las ciudades que visitan estos proveedores.

```
SELECT    UNIQUE P #, CIUDAD
FROM      SP, S
WHERE     SP.S# = S.S#
```

Se obtiene:

| P # | CIUDAD |
|---|---|
| P1 | Talca |
| P1 | Rancagua |
| P2 | Talca |
| P2 | Rancagua |
| P3 | Talca |
| P4 | Talca |
| P5 | Talca |
| P6 | Talca |

En el ejemplo previo, se muestra una reunión en SQL, se pueden dar varios nombres de tablas en la cláusula de la sintaxis FROM, para buscar la información que nos interesa. Los nombres de las tablas en el SELECT y en WHERE se especifican para una mayor claridad.

## CLASE 10

f) Recuperación que implica una reunión de una tabla consigo misma: por ejemplo obtener las parejas de números de proveedores de tal manera que los proveedores estén localizados en la misma ciudad.

```
SELECT    PRIMERO. S #, SEGUNDO. S #
FROM      S PRIMERO, S SEGUNDO
WHERE     PRIMERO. CIUDAD = SEGUNDO. CIUDAD
AND       PRIMERO. S# < SEGUNDO. S#
```

Se obtiene:

| S # | S # |
|---|---|
| S1<br>S2 | S4<br>S3 |

En el ejemplo se describe una consulta a una tabla S, la condición que se requiere es que las ciudades sean la misma y que el primer número del proveedor sea menor que el segundo. La tabla S aparece dos veces en la cláusula FROM para distinguir un proveedor del otro y es por eso que se utilizan los nombres primero y segundo para que se entienda de mejor manera. El hecho que se pida que uno sea menor que otro es para que no aparezcan parejas iguales (S1,S1).

g) Recuperación usando ANY: esta cláusula significa "cualquiera". Obtener los nombres de empleados donde la edad sea igual a 30 años.

```
SELECT    NOMS
FROM      S
WHERE     S # = ANY ( SELECT  S #
                      FROM    SP
                      WHERE   E# = "30")
```

Se obtiene:

| NOMS |
|---|
| MEZA<br>GAJARDO<br>CASTILLO<br>ROMERO<br>SOTO |

En este ejemplo la consulta se extrae de las tablas S y SP porque de las dos se define el resultado, la condición que la edad sea igual a 30 años debe satisfacer para cualquier (ANY) empleados, en este caso lo que aparece en paréntesis equivale a una subconsulta, donde se toma el valor del conjunto total de empleados que corresponda al valor = "30" en la tabla SP, de todo el conjunto (S1,S2,S3....)

Tomará el valor verdadero si y solo si S# tiene uno de los valores S1,S2,S3... y dará el resultado que se solicita.

Observación: la cláusula IN tiene la misma función de ANY, cada una puede compararse al operador Є de pertenencia a un conjunto.

h) Recuperación usando ALL: por ejemplo obtener los nombres de empleados que no tengan 30 años de edad.

```
SELECT    NOMS
FROM      S
WHERE     "30"  ¬ = ALL (SELECT  E#
                         FROM    SP
                         WHERE   S # = S.S #)
```

Se obtiene:

| NOMS |
|---|
| MARCELA |

En el ejemplo se pide la condición que se desplieguen aquellos empleados que no tengan 30 años de edad. La cláusula ALL se define análogamente a una comparación ANY, pero la cláusula ¬ = ALL quiere decir lo contrario que no sea igual a. También puede usarse la cláusula NOT IN. El resultado será dependiendo si solo si se cumple la condición de que haya un empleado que su edad sea no igual "30".

Así como ANY puede escribirse como IN, del mismo modo ¬ = ALL puede escribirse como NOT IN.

i) Recuperación usando EXISTS: por ejemplo obtener los nombres de empleados que tengan la edad de 27 años (igual que el ejemplo anterior) .

```
SELECT    NOMS
FROM      S
WHERE     EXISTS
          (SELECT    *
           FROM      SP
           WHERE     S# = S.S
           AND       E# = "27")
```

Se obtiene:

| NOMS |
|---|
| Claudio |
| Andrea |
| Javier |
| Sandra |

En el ejemplo la cláusula EXIST representa un cuantificador existencial, el valor será verdadero, si solo si, existe a lo menos un registro en la tabla que cumpla la condición. Aquí se desplegaron todos los nombres que cumplían la condición de desplegar todos los nombres de empleados que tuvieran la edad de 27 años, por lo tanto la consulta es para dos tablas S y SP, donde los datos que están en S deben coincidir con la tabla SP. Primero se busca la condición que tenga la información y luego se eligen y despliegan. El primer nombre en la lista es Claudio, el valor correspondiente a la tabla S# es S1, existe un registro de SP con S# igual a S1 y E# igual a "27", si es así entonces Claudio es uno de los valores recuperados, el primero

Para la cláusula NOT EXISTS describe el significado contrario, por ejemplo realizar una consulta que no cumpla la condición. Realizar la condición de preguntar que despliegue los proveedores que no trabajen el producto de oficinas, debería desplegar una lista con todos los proveedores que no trabajen en ese rubro.

# CLASE 11

## 3.1. Funciones Integradas:

El lenguaje de recuperación que hasta ahora hemos estudiado, resulta ser inadecuado para algunos casos. Una simple consulta como ¿Cuántos empleados hay? , no se puede expresar con las cláusulas vistas de SQL anteriormente. Por lo tanto existen algunas funciones integradas especiales para resolver estos problemas.

Las funciones son:

| | |
|---|---|
| COUNT: | EL NÚMERO DE VALORES |
| SUM: | LA SUMA DE LOS VALORES |
| AVG: | EL PROMEDIO DE LOS VALORES |
| MAX: | EL VALOR MAS GRANDE |
| MIN: | EL VALOR MAS PEQUEÑO |

Para SUM y AVG, la columna debe tener valores numéricos. En general, el argumento de la función puede estar establecida por la cláusula UNIQUE, para indicar que los valores redundantes deben eliminarse antes de que la función se aplique.

Para la función COUNT debe especificarse UNIQUE, sin embargo, la función COUNT (*) se provee para contar todas las filas de una tabla sin eliminar duplicados.

a) Función cláusula SELECT: obtener el número total de empleados.

```
SELECT    COUNT (*)
FROM      S
```

Se obtiene:

| |
|---|
| 10 |

La letra S es el nombre de la Tabla, Lo que el ejemplo pide por condición que, de la tabla S se extraiga el total de empleados, que en este caso es 10, que es el número total que da como resultado. COUNT es un contador.

b) Función SELECT con un predicado: obtener la cantidad total de la parte P2 suministrada.

```
SELECT    SUM (CTD)
FROM      SP
WHERE     P # = "P2"
```

Se obtiene:

| |
|---|
| 1000 |

Lo que se pide en el ejemplo es obtener la cantidad total de P2 que está en la Tabla SP, lo cual corresponde a la suma de los datos numéricos de P2 que como resultado dan 1000.

c) Uso de GROUP BY: para cada parte suministrada, obtener el número de la parte y la cantidad total suministrada de la misma.

```
SELECT    P #, SUM (CTD)
FROM      S
GROUP BY  P #
```

Se obtiene:

| P# | |
|---|---|
| P1 | 600 |
| P2 | 500 |
| P3 | 1400 |
| P4 | 500 |
| P5 | 200 |
| P6 | 410 |

En el ejemplo el operador GROUP BY reordena conceptualmente la tabla en grupos. En el ejemplo la tabla se reordena de manera que el primer grupo tenga las filas para P1, el segundo las filas para P2, una vez realizada la condición que se solicitó.

También para el DML de SQL, incluye tres operaciones de actualización: UPDATE (modifique), INSERT (inserte), DELETE (suprima).

Algunos ejemplos:

a) Actualización de un solo registro: cambie el color de la parte P2 a amarillo y aumente su peso en 5.

```
UPDATE   S
SET      COLOR = "AMARILLO"
         PESO = PESO + 5
WHERE    P# = "P2"
```

Dentro de una cláusula SET, cualquier referencia a un campo a la derecha de un signo igual, se refiere al valor de ese campo antes de que se efectúa la actualización. No es posible actualizar más de una tabla.

Se define en el ejemplo que si P# = "P2" cambiar el color a amarillo y si peso aumente a 5 más. Cambia solamente un sólo registro de P#.

b) Inserción de un registro: por ejemplo adicionar la parte P8 (de nombre “CLAUDIO”, color “ROJO”, ciudad “TALCA”) a la tabla P.

```
INSERT INTO P:
            <”P8”,”CLAUDIO”, “ROJO”, “TALCA”>
```

Se debe destacar que en la operación se especifica en orden de izquierda a derecha de las columnas en la tabla.

También se puede insertar un nuevo registro:

```
INSERT INTO P (P #, EDAD, NOM)
            < “P8”, “35”, “CLAUDIO”>
```

En el ejemplo se muestra, que un registro nuevo de la tabla P se crea con el número, la edad y nombre especificados, para un registro nuevo no se necesita ser el mismo que tienen en la tabla.

c) Suprimir un solo registro: por ejemplo eliminar el proveedor S2

```
DELETE  S
WHERE   S # = “S2”
```

En el ejemplo la condición pide la eliminación en la tabla S del proveedor que sea igual a S2, si en la tabla SP, que es la tabla que tiene otro tipo de información de ese proveedor, hay alguna información del proveedor que se eliminó; por lo tanto habrá una falta para la regla de integridad.

d) Suprimir los registros múltiples: por ejemplo suprimir toda la información de la tabla SP.

```
DELETE  SP
```

En el ejemplo la condición pide que elimine toda información de la tabla SP, entonces SP sigue siendo una tabla, pero vacía.

# CLASE 12

## 4. MANIPULACIÓN DE DATOS.-

La manipulación de datos se refiere a las operaciones de insertar, recuperar, eliminar o modificar datos. Dichas operaciones son realizadas a través del lenguaje de manipulación de datos (DML, Data Manipulation Language), que es el que permite el acceso de los usuarios a los datos.

Existen básicamente 2 tipos de lenguajes de manipulación de datos:

a) Procedimentales: los DML requieren que el usuario especifique qué datos se necesitan y cómo obtenerlos para poder aplicar la sentencia correcta y realizar la consulta.

b) No Procedimentales: los DML requieren que el usuario especifique qué datos se necesitan y sin especificar cómo obtenerlos, dejando todo a cargo del administrador de base de datos.

Como el Lenguaje de Manipulación de Datos consiste en la forma de recuperar datos almacenados en una Base de Datos. El DML también tiene dos perspectivas para utilizar:

1.- Proposiciones integradas: Son una serie de instrucciones contenidas en un programa.

2.- Consulta (Query): Son consultas rápidas a una Base de Datos.

Para ambas se requiere un lenguaje de programación (SQL). Entonces, el DBMS se convierte en un intermediario entre el usuario y la Base de Datos almacenada y a través de un programa o una consulta se puede extraer información.

El lenguaje de Manipulación de Datos utiliza uno de estos tres operadores para poder realizar las consultas a la base de datos. A continuación se explicarán[2]:

[2] Según James Senn.

## a) Operaciones SELECT:

Produce una nueva tabla, en respuesta a alguna consulta o solicitud creada a partir de las filas de la tabla inicial que cumplan con los criterios de la solicitud.

¿Cuáles son los nombres de los clientes que hicieron pedidos el 24/12? Al examinar las filas de la relación CLIENTE, donde la fecha del pedido tenga como valor 24/12. Entonces se crea una nueva tabla que tenga dos filas más y dos columnas. Sólo se incluyen los campos que se han solicitado.

## b) Operaciones PROJECT:

Aquí se crea una nueva tabla a partir de los datos extraídos, escoge columnas de una tabla ya existente.

¿Cuáles son los números de cuenta de los clientes que arriendan sillas y mesas?

| Número de cuenta |
|---|
| 812 |
| 976 |
| 1054 |
| 1950 |
| 2200 |

## c) Operaciones JOIN:

Crea una nueva relación combinando dos tablas existentes, eligiendo los registros que cumplan con los criterios solicitados en las preguntas y removiendo después los registros duplicados.

¿Cuáles son los nombres y los números de cuentas de aquellos clientes que solicitan vajillas?

| Nuevo | |
|---|---|
| Marcela | 568 |
| Claudio | 658 |
| Andrés | 1250 |
| Patricio | 2450 |

Estos operadores son la base de todas las operaciones que se puedan realizar con los datos. Con estas operaciones se pueden realizar adiciones, borrados y cambios.

A continuación, se analizarán cada una de ellas.

### Adición:

Se añade algún dato sin afectar a los demás registros.

INSERT INTO CLIENTE VALUES
“1245”, “Claudio Meza”, “4 oriente 2064, Talca”

### Borrado:

Se borra la fila del registro que se ha especificado.

DELETE FROM CLIENTE WHERE CLIENTE-ID = 254

### Cambio:

Se busca el registro que se desea cambiar con la llave especificada y los valores de los datos se cambian con los nuevos datos.

UPDATE CLIENTE
SET DIRECCIÓN = “18 PONIENTE # 237, TALCA” WHERE CLIENTE-ID =254

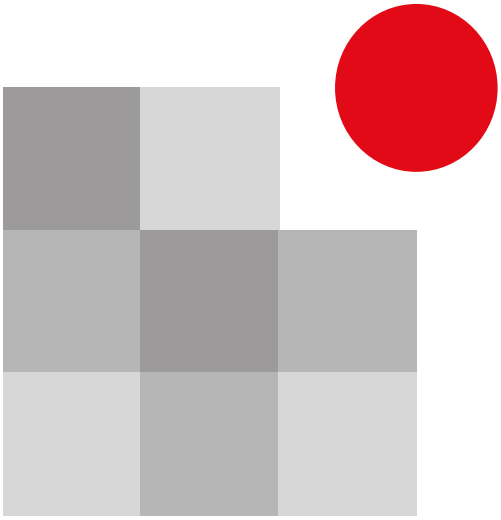
En la siguiente figura se presentará un resumen de las operaciones relacionales.

Figura N° 6: Resumen Operaciones Relacionales Manipulación de datos

| OPERADOR RELACIONAL | DESCRIPCIÓN DE LA OPERACIÓN |
|---|---|
| SELECT | Crea una nueva relación sacando filas que cumplan con los criterios establecidos |
| PROJECT | Crea una nueva tabla sacando columnas con los criterios que se pidan |
| JOIN | Crea una nueva relación a partir de las filas de dos tablas que tengan atributos que cumplan los criterios buscados. |

Realice Ejercicios N° 17 al 30

SISTEMAS DE INFORMACIÓN II



IPLACEX
instituto profesional

UNIDAD II

# MODELANDO DATOS CON ESQUEMA ENTIDAD - RELACION

# CLASE 01

## 1. INTRODUCCIÓN MODELAMIENTO DE DATOS

En palabras muy sencillas se podría decir que existe el mundo real y el mundo de las ideas, los problemas están en el mundo real, pero ingresan en el mundo de las ideas, en donde se piensan y solucionan. Una vez que se ha encontrado la solución a los problemas se aplica al mundo real.

Existen varios procesos, los cuales se explican a continuación:

a) Proceso de Abstracción: corresponde al proceso de captura de problemas del mundo real para solucionarlo en el mundo de las ideas.

b) Proceso de Concreción: es el proceso por el cual se aplica (se concreta) en el mundo real la solución o las soluciones alternativas del problema emanada del mundo de las ideas.

c) El Proceso de Modelación: tiene más que ver con un proceso de abstracción, se observa una realidad física de una empresa, un problema, y se decide buscar una solución en el mundo de las ideas.

En resumen, se puede señalar que el proceso de modelación, "es un proceso de abstracción basado en algún modelo de datos".

Quizás surja la duda de lo que es un modelamiento, por lo cual se procederá a otorgar su definición.

Modelamiento: es un proceso de abstracción apoyada en alguna herramienta que nos ayude a obtener una representación lo más fiel posible de la realidad que se quiere recoger.

El modelo es el que nos provee del marco y los conceptos que nos permiten recoger esta realidad.

Como una forma de recordar lo visto en la unidad anterior, se definirá el concepto de modelo de datos.

Los modelos de datos son una herramienta intelectual que nos provee de una interpretación de la realidad que se aspira recoger.

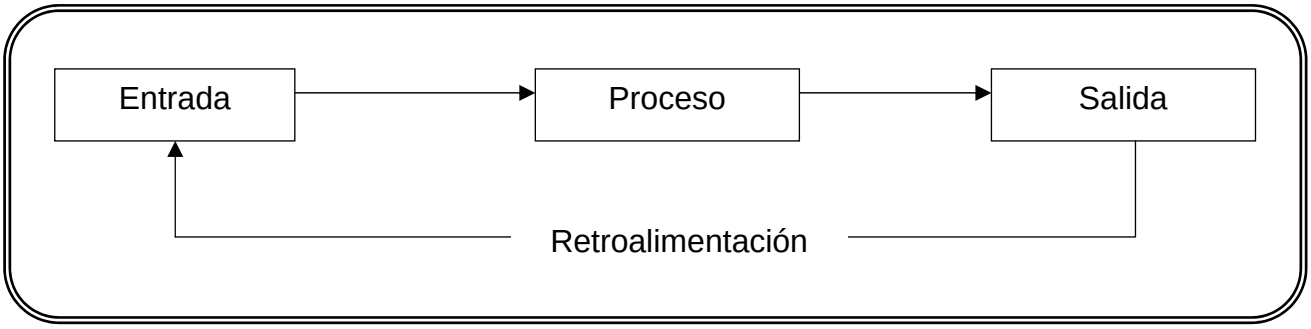
Es el dispositivo de abstracción que nos permite ver el bosque en contraposición a los árboles.

Se obtiene más que datos, información.

Un dato es una cadena de caracteres, la información es una cadena de caracteres con sentido, con significado.

Un modelo de datos se puede graficar del siguiente modo:

Figura N° 1: Modelo de Datos




## 1.1. Definiciones Básicas

El modelo Entidad – Relación (E-R) está basado en una percepción de la vida real que consta de una colección de objetos o cosas básicas, tanto concretos como abstractos, en donde el MER los denomina entidades. Este modelo fue propuesto por Peter Chen[1] en 1976 buscando una visión natural de la realidad consistente en entidades y relaciones.

El modelo entidad-relación es un modelo basado en 4 conceptos: entidad, relación, atributo y dominio.

A continuación, se procederá a definir cada uno de ellos.

a) Entidad: es todo objeto concreto o abstracto de interés de la realidad que se desea recoger, pueden ser personas, bienes físicos o abstractos.

   Por ejemplo:

   Persona, vivienda, vehículos, banco.

b) Relación: es una asociación entre dos o más entidades que no necesariamente son distintas.

[1] En 1976 Peter Chen introdujo el llamado modelo de Entidades y Relaciones (ER). Es el modelo conceptual más utilizado para el diseño conceptual de bases de datos.

Por ejemplo:

1) Juan Soto, alumno, entra a este sistema curricular, la relación es la inscripción.
2) Los alumnos del colegio “Pablo Neruda” tienen clases regidas por un horario y sala fija, la relación es la asistencia, en donde participan los alumnos, sala y horario.

En la práctica las relaciones asocian dos entidades y aquellas que asocian más suelen ser reducidas a dos.

Una relación es una asociación entre dos o más entidades, por ejemplo:

| | |
|---|---|
| Relación | : Clase |
| Entidades participantes | : Profesor, Sala, Curso |
| Ocurrencia | : Clase1 |
| Conjunto de Ocurrencias | : Clase1, Clase2, Clase3... |

c) Atributo: es toda característica o propiedad de una entidad o relación que nos interese representar.

Por ejemplo:

En el caso de un año de matrícula, los atributos de los alumnos son: nombre, fono, fecha de nacimiento, etc. La fecha del matrimonio es un atributo de la relación denominada matrimonio.

d) Dominio: el conjunto de valores que puede asumir un atributo en su dominio.

Por ejemplo:

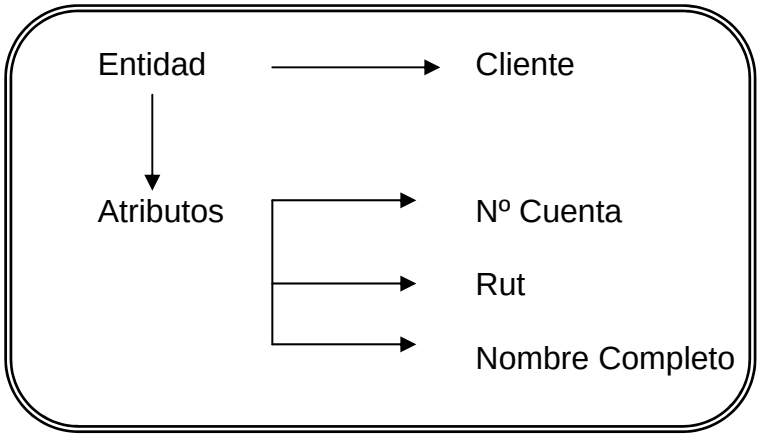
Si A es un atributo, E una entidad, R una relación y D1, D2, D3... Dn son sus dominios, entonces se tiene:

$$A : E \text{ ó } R \Longrightarrow D_0, D_1, D_2, D_3 \ldots D_n$$

Las estructuras básicas que utiliza el Modelo E-R son las entidades y las relaciones que pasaremos a determinar:

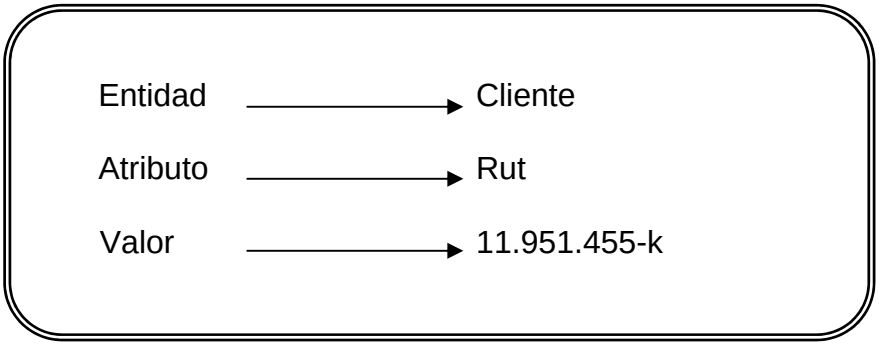
Ejemplo Nº 1

Cada entidad tendrá un valor para cada atributo




Ejemplo Nº 2

Resumen




Una ocurrencia de la entidad persona sería Juan, Andrés, etc.

Ejemplo Nº 3

| | |
|---|---|
| Entidad | : Persona |
| Ocurrencia | : Pablo |
| Conjunto de ocurrencias | : Pedro, Diego, Claudia |

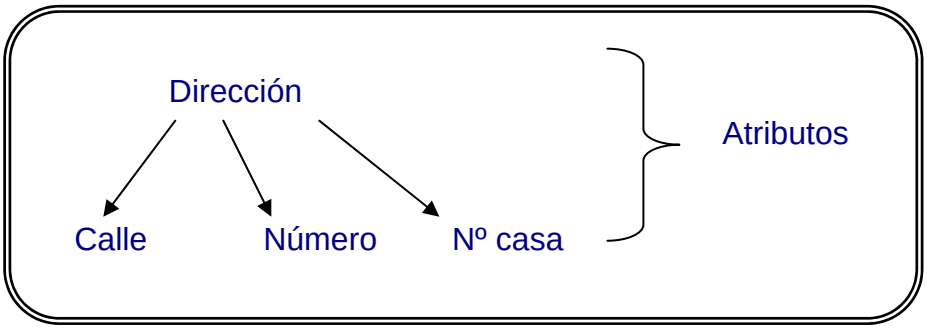
Este conjunto de valores sobre el cual se aplica un atributo se denomina Dominio.

## CLASE 02

Existen distintos tipos de Atributos: "simples" o "compuestos", "monovaluados" o "multivaluados", "almacenados o derivados". A continuación, se explicará cada uno de ellos:

a) Atributos Compuestos o Simples: los atributos compuestos son aquellos que pueden dividir sus componentes en más pequeños, que también representan atributos.

Ejemplo Nº 4




Aquellos atributos que no son divisibles son atributos simples, ejemplo: Rut.

El atributo compuesto será tal siempre y cuando sea necesario para un usuario, sino quedará como atributo simple.

b) Atributos Monovaluados o Multivaluados: los atributos monovaluados son aquellos en donde el atributo tiene un solo valor, como por ejemplo el rut de una persona.

Hay algunos casos en que el atributo puede tener un conjunto de valores para la misma entidad, como por ejemplo, la manzana que es una entidad en que el atributo color tiene dos valores: roja y verde. También los nombres de una persona que pueden ser más de uno.

Este tipo de atributos se denominan "multivaluados"

c) Atributos Almacenados o Derivados: algunos atributos se pueden relacionar con dos valores. Como por ejemplo el Nº de empleados que trabaja en un determinado departamento de una tienda; si se considera el Nº de empleados que trabaja en ese departamento, o bien, los atributos edad y fecha de nacimiento de la entidad persona. El valor de la edad se puede adquirir con la fecha actual y el valor de la fecha de nacimiento de la persona.

Entonces se dice que la edad es un atributo "derivado" y es derivable de la fecha de nacimiento que vendría siendo un atributo "almacenado".

En algunos casos una entidad podría no tener ningún valor aplicable para un atributo, en estos casos "los valores son nulos".

Ejemplo Nº 5

No todas las personas tienen un título universitario, por lo tanto, la entidad persona tendría como valor "nulo" en el atributo título universitario

Realice ejercicios nº 1 al nº 5

## 1.2. Tipos de Entidad y Claves.

Entre todos los atributos de un tipo de entidad han de existir uno o varios atributos (simples y/o compuestos) que identifiquen particularmente cada uno de los ejemplares de ese tipo de entidad.

Cada uno de estos conjuntos de atributos se denomina Identificador Candidato (IC).

Todo IC debe cumplir la condición de ser unívoco y mínimo, cuando un IC es compuesto, el número de los atributos que lo componen debe ser mínimo, en el sentido de que la eliminación de cualquiera de ellos le haría perder su carácter de identificador. Todos los atributos identifican a la entidad, pero debe haber uno que permita acceder a la entidad en forma única para que no haya redundancia de datos. Ese atributo es el IC. Si se pierde el IC, no hay manera de acceder a la información sin que haya error de redundancia de datos.

Un ejemplo es el rut, que identifica a cada persona que existe, nunca habrá una persona con el mismo rut. Pero puede haber alguien que no tenga rut, y otro identificador candidato podría ser, en caso de algún alumno, el número de matricula, o el nombre completo, algo que lo identifique de las demás entidades.

Entre los IC se elige uno como Identificador Principal (IP) y el resto serán Identificadores Alternativos (IA).

Generalmente una base de datos tiene grupos de entidades similares.

### Ejemplo Nº 6

Una tienda que quiere sumar clientes almacenará la misma información para los nuevos clientes usando como ejemplo los datos que tiene de los ya existentes en su base de datos. Estas entidades tendrán lógicamente los mismos atributos, pero con distintos valores.

Cada tipo de entidad se describe por su nombre y atributos.

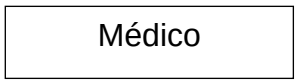
Ejemplo Nº 7

| NOMBRE DEL TIPO | | |
|---|---|---|
| ENTIDADES | CLIENTE | TIENDA |
| CONJUNTO DE ENTIDADES | Nombre, Rut, Nº Cuenta | Nombre , Ubicación, Gerente ventas |
| | Juan, 11.951.455-k, 26 | A. Paris, Talca, Javier R. B. |
| | Claudio, 13.305.019-1, 531 | Falabella, Rancagua, Andrés F. |
| | Andrea, 12.625.896-3, 256 | Ripley, Rancagua, Juan A. |

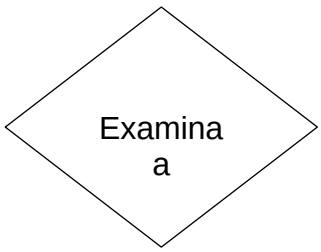
En el ejemplo se puede observar dos entidades Cliente y Tienda, y cada tipo con sus conjuntos de entidades y sus valores respectivos.

Una de las ventajas del modelo E-R es su “capacidad gráfica” (simbología).

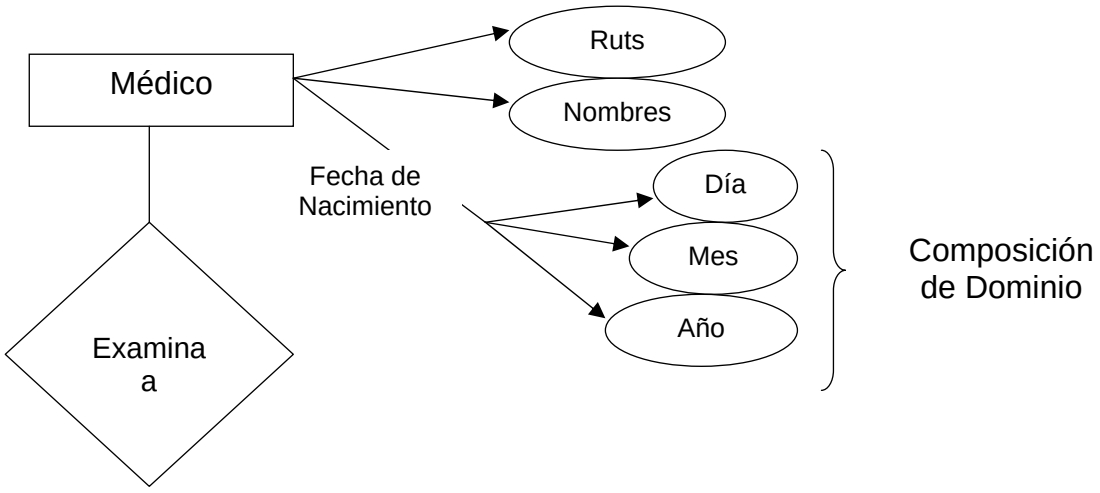
a) Las entidades se representan por un rectángulo.




b) Las relaciones con un rombo.



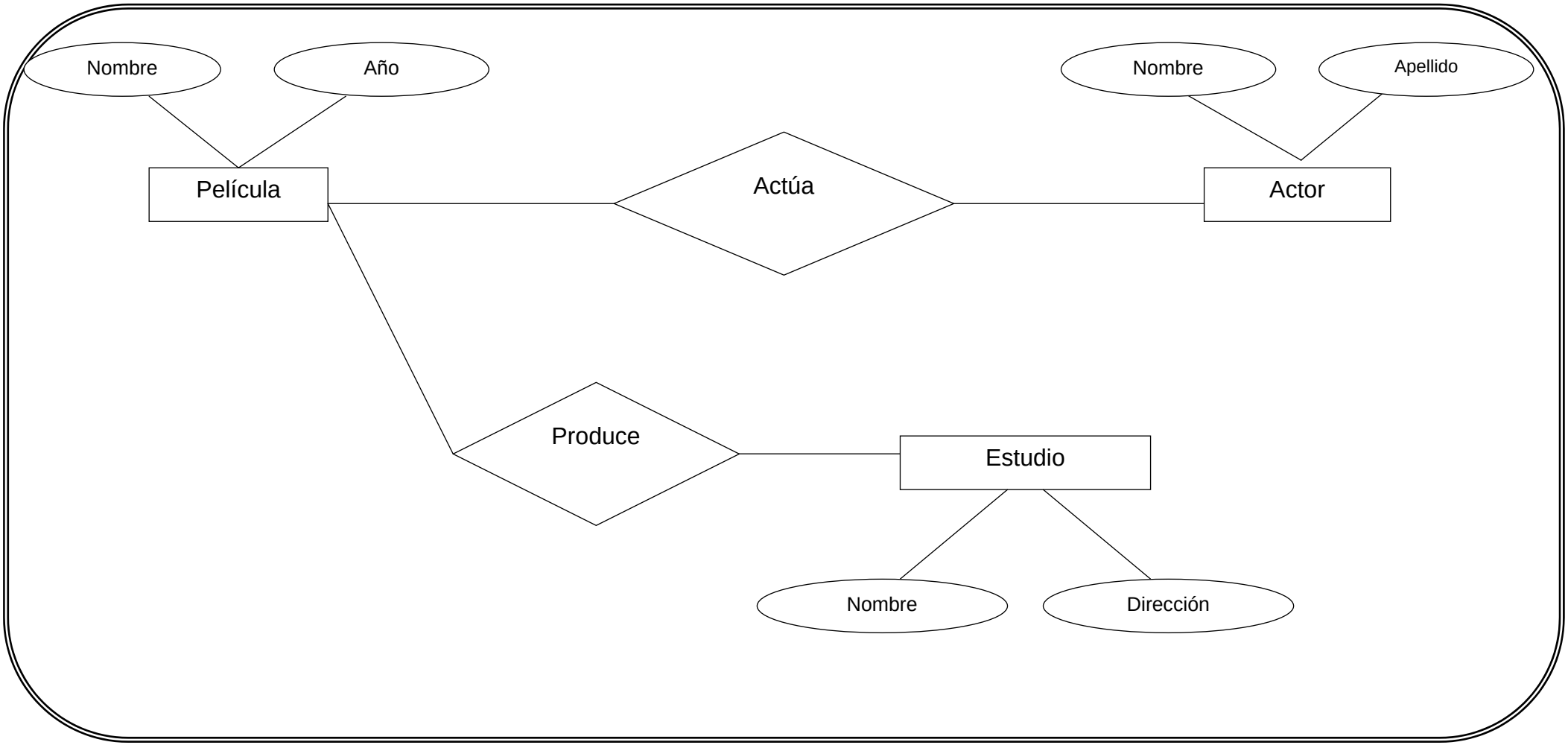
c) Los dominios en un círculo, éste se pone en una “composición de dominio”.



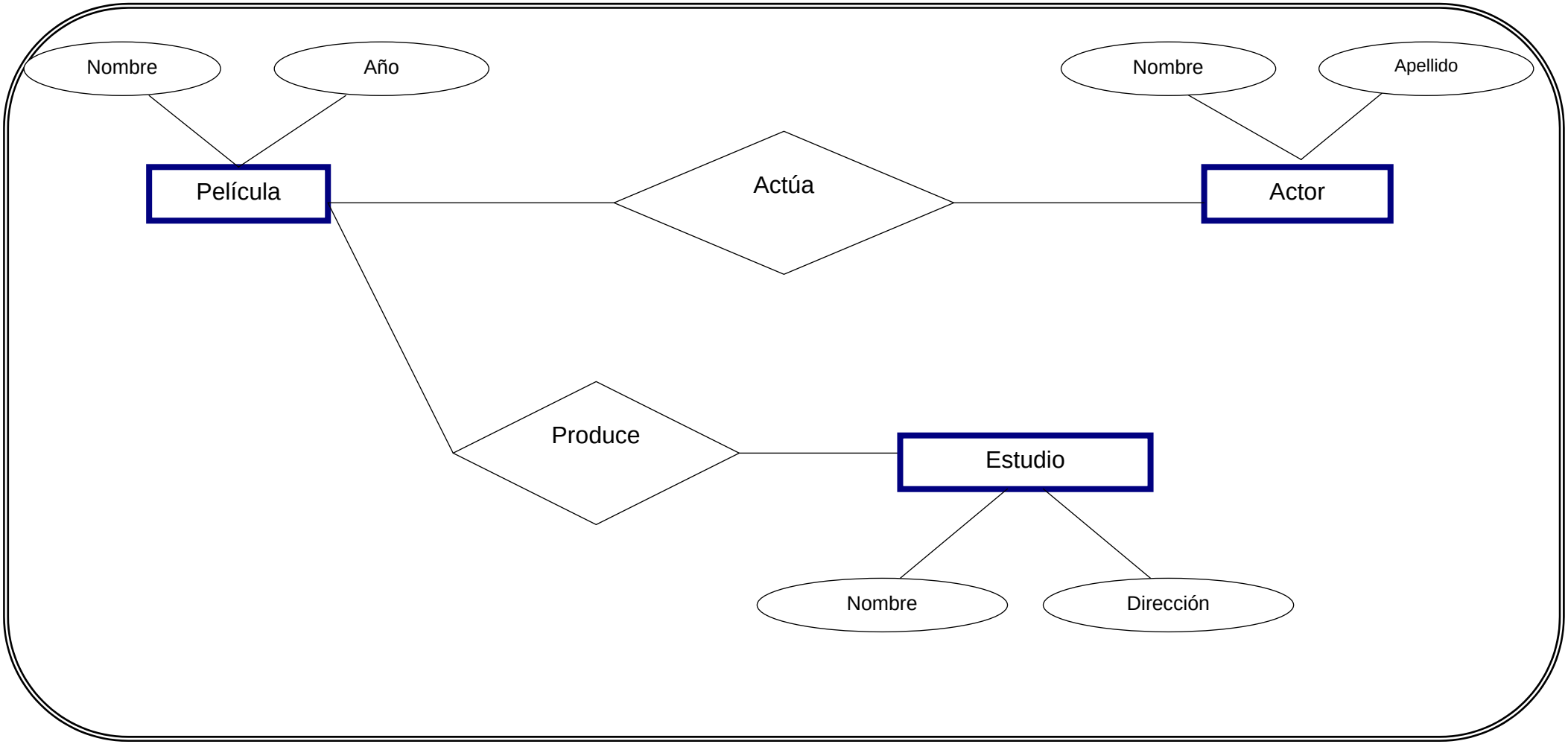

d) Los atributos se representan por una flecha.

# CLASE 03

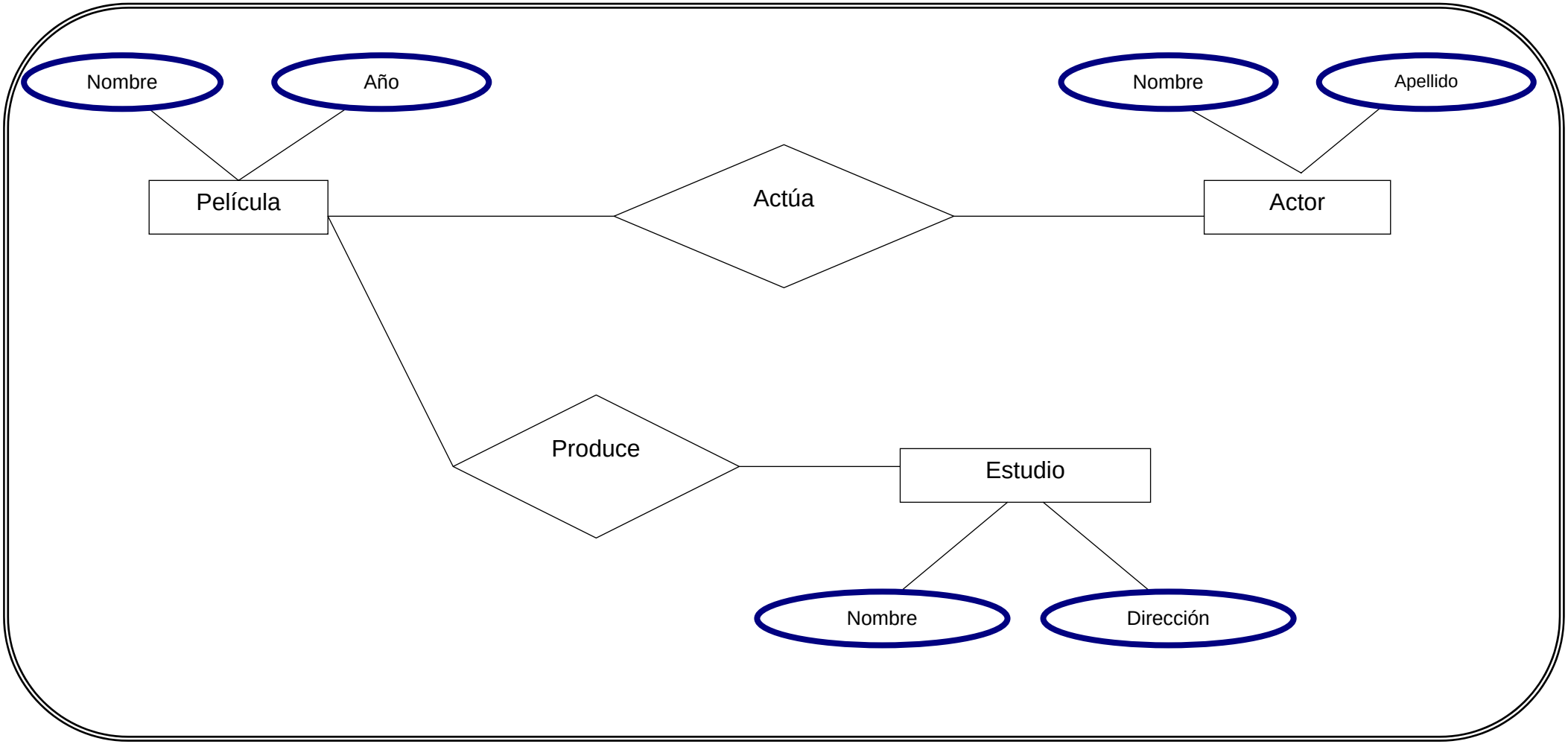
## Ejemplo Diagrama Entidad Relación



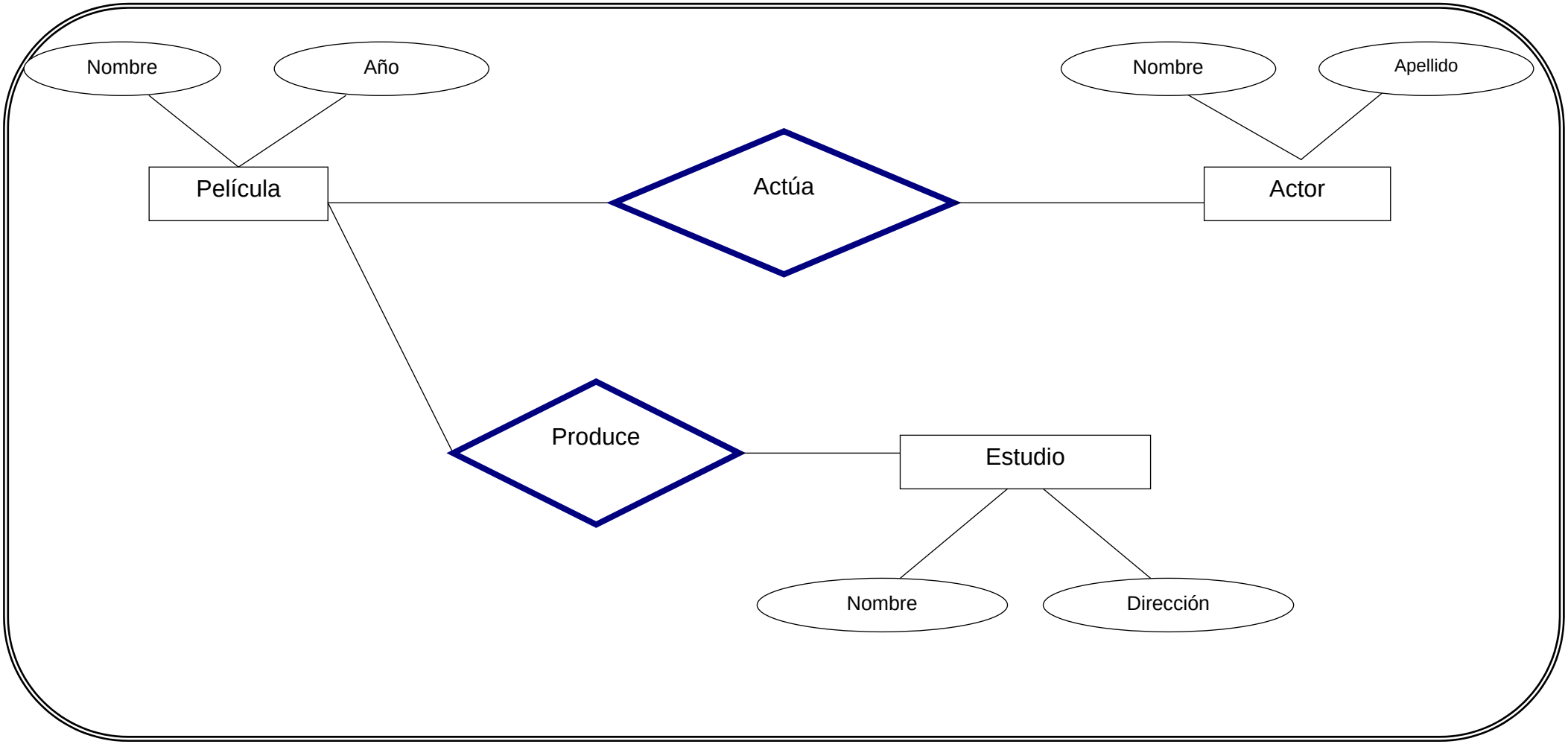
# Entidades


Nombre
Año
Nombre
Apellido
Película
Actúa
Actor
Produce
Estudio
Nombre
Dirección

Atributos


Nombre
Año
Nombre
Apellido
Película
Actúa
Actor
Produce
Estudio
Nombre
Dirección

# Relaciones


Nombre
Año
Nombre
Apellido
Película
Actúa
Actor
Produce
Estudio
Nombre
Dirección

Una entidad siempre tiene atributos cuyos valores son distintos para cada entidad, como por ejemplo el rut de una persona. Éstos se denominan "Atributos Clave" y sus valores sirven para identificar a las entidades de manera única.

También existen los atributos compuestos que son aquellos en donde varios atributos juntos constituyen una clave, es decir, que los valores de los atributos son distintos para cada entidad individual, el cual se convertiría en el atributo clave.

El hecho de que haya un atributo clave quiere decir que se debe cumplir la propiedad de unicidad, es decir, se prohíbe tener a dos atributos el mismo valor para el atributo clave. Es una restricción para toda entidad.

Por Ejemplo el Rut de una persona, jamás habrá dos personas con el mismo Rut, o un vehículo con la patente igual a otro vehículo.

Algunos tipos de entidad tienen más de un atributo clave, como por ejemplo, los datos de un cliente de una tienda para poder acceder a los datos de él se requiere del Rut y Nº de Cuenta.

Existe otro concepto: "clave primaria" que se usará para mostrar una clave candidata que es elegida por el diseñador de la base de datos como elemento principal para identificar las entidades y poder acceder a la información.

## CLASE 04

### 1.3. Conjuntos De Relaciones.

Como ya se había mencionado una "relación" es una asociación entre diferentes entidades. Por ejemplo, se puede definir una relación que asocie a un cliente Meza con el Nº de cuenta 255. Esto especifica que Meza es un cliente con el Nº de cuenta 255.

Por ejemplo:

Consideremos los dos conjuntos de entidades cliente y nº de cuenta, se puede definir el conjunto de relaciones cliente – cuenta para denotar la asociación entre un cliente y el nº de cuenta que corresponde al cliente al momento de adquirir algún producto.

La asociación entre los conjuntos de entidades se define como participación, es decir, los conjuntos $E_1$, $E_2$..., $E_n$, participan en el conjunto de relaciones R.

Una relación representa en un esquema E-R una asociación entre entidades en el desarrollo del mundo real que se desenvuelve. Como por ejemplo: el Sr. Carlos Leiva, cliente, a través de su número de cuenta, puede acceder a comprar las zapatillas que tanto ansiaba por intermedio del crédito obtenido.

La función que cumple una entidad en una relación se llama "papel de la entidad". Cada entidad sabe qué función cumple y cuál es el significado que tiene dentro de una relación. Los conjuntos de entidades que participan en el conjunto de relaciones son distintos, por lo tanto, los papeles están incluidos y no se especifican habitualmente. Sin embargo, son válidos cuando el significado de una relación necesita aclaración.

## 1.4. Correspondencia De Cardinalidades.

Cardinalidad se define como el número de veces con que una ocurrencia de una entidad participa en una relación.

### Ejemplo Nº 8

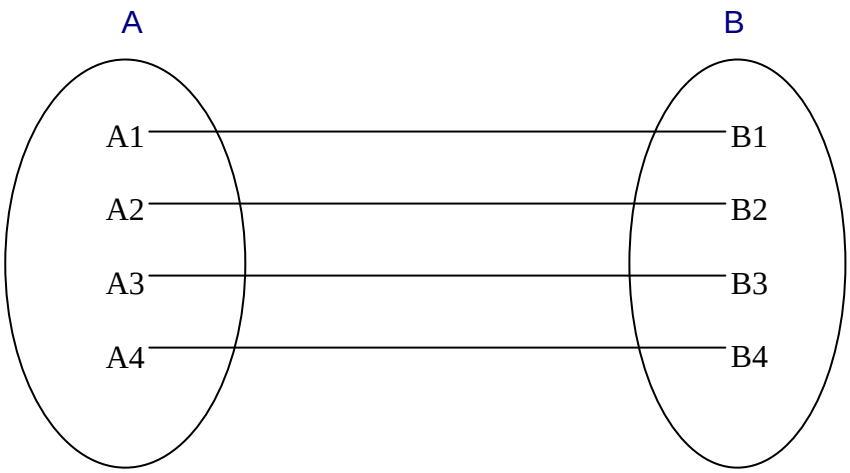
Por ejemplo, si se asume que cada persona al viajar tiene solo un pasaje para llegar al destino la cardinalidad de la relación sería uno en cada dirección.


La cardinalidad es la más útil describiendo conjuntos de relaciones, sin embargo, a veces contribuye a la descripción de conjuntos de relaciones que implican más de dos conjuntos de entidades.

Realice ejercicios nº 6 al nº 10

Para un conjunto de relaciones entre los conjuntos de entidades A y B, la cardinalidad debe ser una de las siguientes:

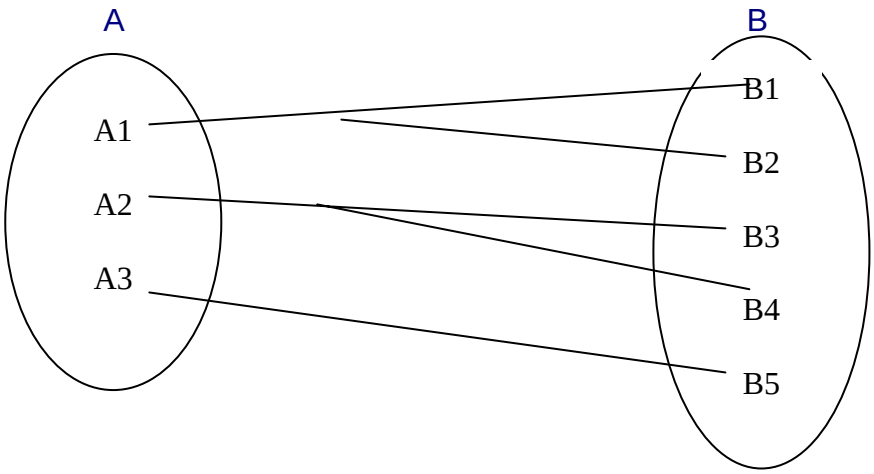
a) Uno a uno. una entidad en A se asocia con una entidad en B, y una entidad en B se asocia con una entidad en A.




Ejemplo: En Chile un marido tiene una esposa o bien una esposa tiene un marido.

Notación = 1:1

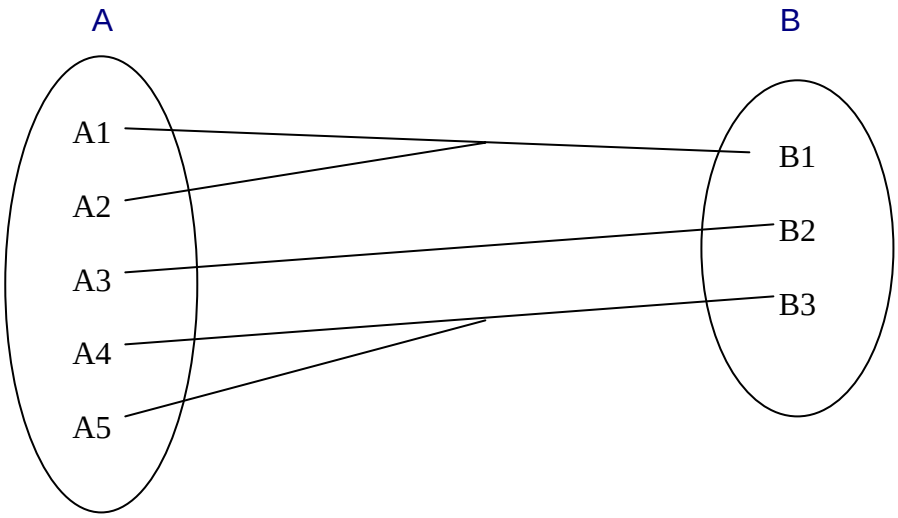
b) Uno a varios: una entidad en A se asocia con cualquier número de entidades en B. Una entidad en B, se puede asociar a lo sumo con una entidad en A.




Ejemplo: Un departamento tiene muchos empleados, un empleado está en un sólo departamento.

Notación =1:N

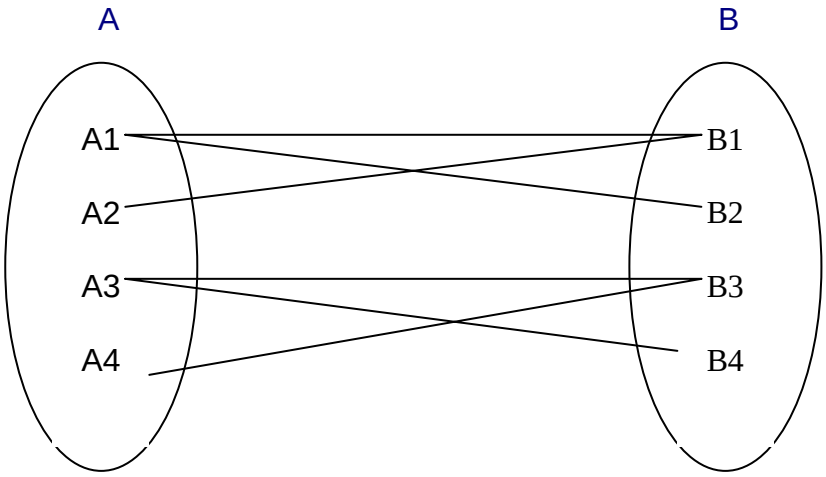
c) Varios a Uno: una entidad en A se asocia con una entidad en B. Una entidad en B, sin embargo, se puede asociar con cualquier número de entidades en A.




Ejemplo: Un pasajero va a un solo destino, y a un solo destino pueden ir varios pasajeros

Notación: N:1

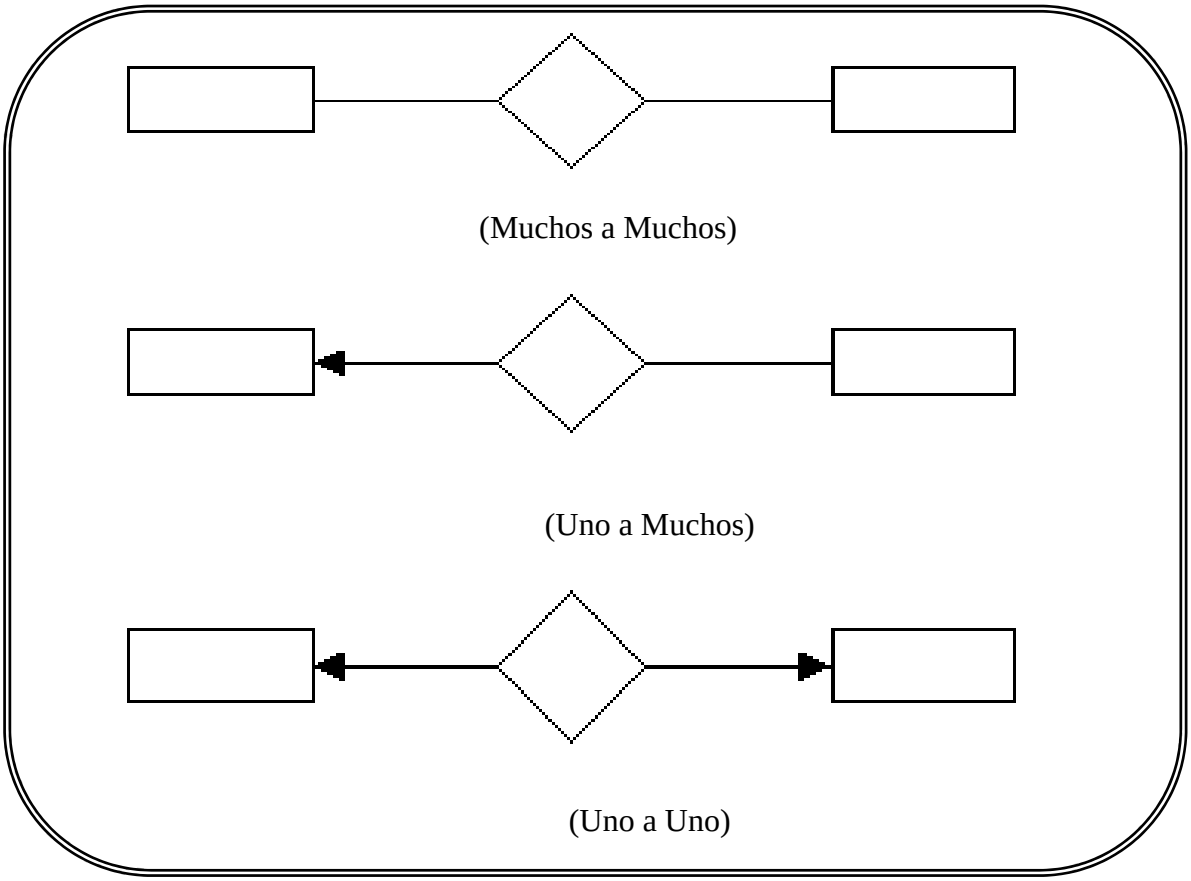
d) Varios a varios: una entidad en A se asocia con cualquier número de entidades en B, y una entidad en B se asocia con cualquier entidad en A.




Ejemplo: Un estudiante sigue muchos cursos, un curso tiene muchos estudiantes.

Notación: N:N

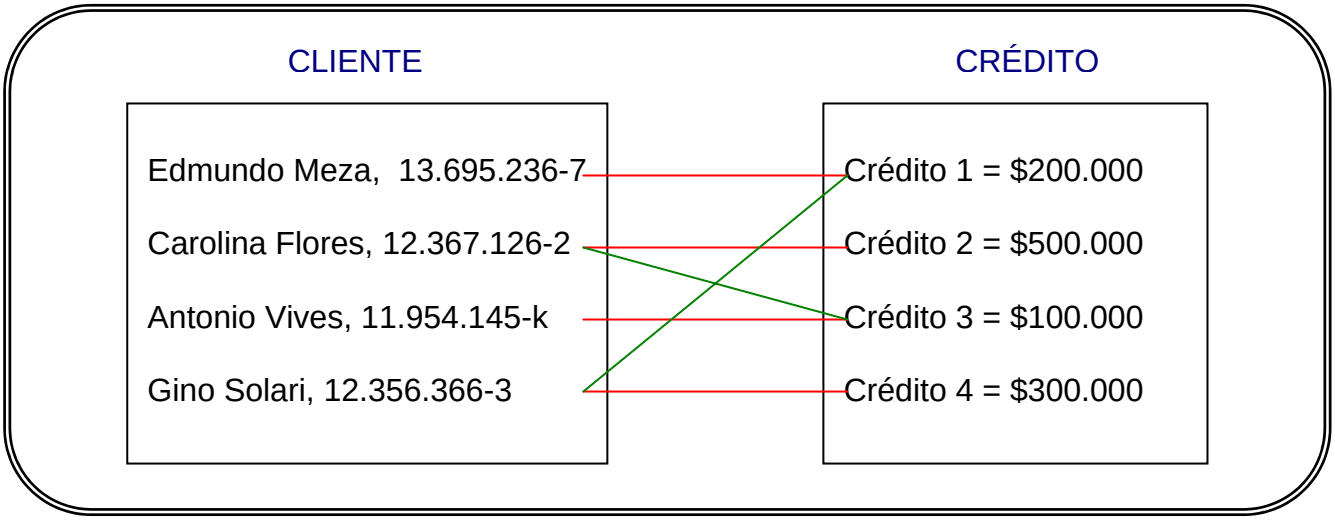
Ejemplo Nº 9




En el ejemplo se observa la simbología para representar las cardinalidades.

La cardinalidad para un conjunto de relaciones particular es obviamente dependiente de la situación del mundo real que el conjunto de relaciones modela, el cual va a depender de la situación que se manifiesta para representar el modelo.

Ejemplo Nº 10




En el ejemplo, el conjunto de relaciones de solicitar un crédito en el banco puede pertenecer únicamente a un cliente y un cliente puede tener varios créditos, por lo tanto la relación cliente a créditos es uno es a varios.

A continuación se presenta un cuadro resumen según cardinalidad, se pueden clasificar las relaciones de los siguientes tipos:

| TIPO | RELACIÓN | REPRESENTACIÓN |
|---|---|---|
| 1:1 | Una a una: la cardinalidad máxima en ambas direcciones es 1. | 1 — ◇ — 1 |
| 1:N | Una a muchas: la cardinalidad máxima en una dirección es 1 y en la otra muchos. | 1 — ◇ — N |
| N:M | Muchas a muchas: la cardinalidad máxima en ambas direcciones es muchos. | N — ◇ — M |

# CLASE 05

## 2. CARDINALIDAD MÁXIMA Y MÍNIMA

Aunque normalmente se trabaja sólo en la máxima cardinalidad, a veces es útil especificar la cardinalidad mínima.

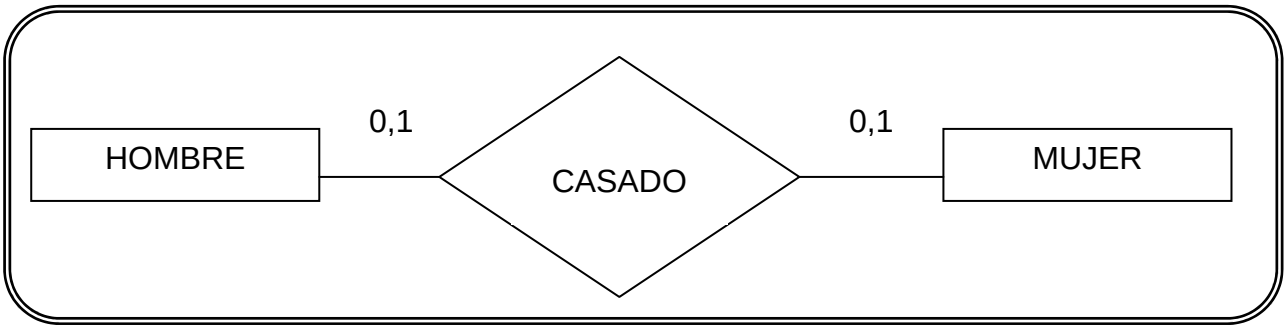
Por lo cual, a continuación se procederá a definir cada una de ellas.

a) Cardinalidad Mínima: es el número mínimo de veces con que una ocurrencia de una entidad participa en una relación determinada.

b) Cardinalidad Máxima: es el número de veces con que una ocurrencia de una entidad participa en la relación.

Por ejemplo tenemos la relación CASADO que existe entre los conjuntos Hombre y Mujer, puesto que muchos hombres y mujeres no están casados entonces la cardinalidad mínima será cero para ambos lados. Se escribirá “0,1” más cerca del conjunto Mujer para indicar que un hombre está casado con cero o una mujer. Inversamente el “0,1” cercano al conjunto Hombre, para indicar que una mujer está casada con cero o un hombre.

Representando las cardinalidades mínimas y máximas quedaría:




Cardinalidad mínima = 0
Cardinalidad máxima =1

De otro modo:

¿Puede haber un hombre no casado? Si.

Cardinalidad Mínima = 0

¿Puede un hombre estar casado con más de una mujer? No.

Cardinalidad Máxima = 1

Se lee: Todo hombre está casado con 0 ó 1 mujer, o bien toda mujer está casada con 0 ó 1 hombre. Los diagramas de relación se pueden leer de izquierda a derecha, y de derecha a izquierda.

## 2.1. Dependencias de Existencias.

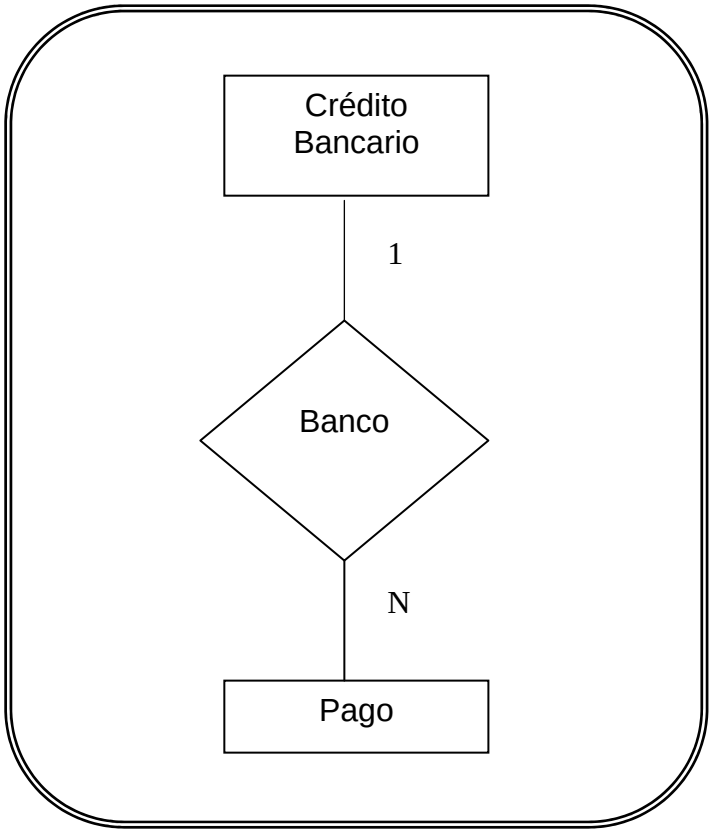
Una clase importante de dependencias son las "dependencias de existencias". Si la existencia de la entidad X depende de la existencia de la entidad Y, entonces se dice que X tiene dependencia de existencia de Y. Si por alguna operación se borra Y, también se borrará X y recibiría el nombre de "Entidad subordinada".

### Ejemplo Nº 11

Si tenemos una entidad Crédito-bancario y el conjunto de entidades Pago que mantiene información acerca de todos los pagos que se han hecho al crédito-bancario. La cardinalidad que se forma es uno es varios, desde crédito-bancario a pago. Cada entidad Pago debe estar asociada con una entidad Crédito-bancario.

Si una entidad Crédito-bancario se borra, todas sus entidades Pago se deben borrar también. En cambio, las entidades Pago se pueden borrar de la base de datos sin afectar a la entidad Crédito-bancario. Entonces esta entidad es dominante y la entidad Pago es subordinada.

A continuación, se representará el ejemplo en forma gráfica.



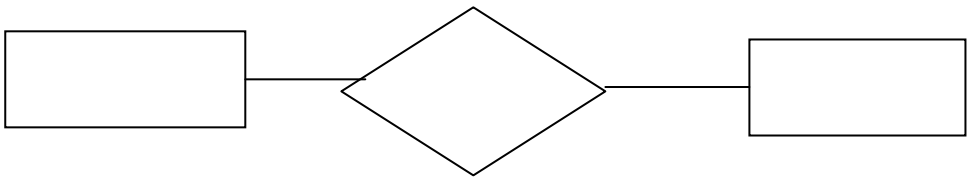

## 2.2. Diagrama Entidad - Relación.

La estructura general de una base de datos se puede representar gráficamente mediante un diagrama E-R. La simplicidad de esta técnica y su claridad puede ser en gran parte el uso del modelo E-R. Tal diagrama consta de los siguientes componentes principales:

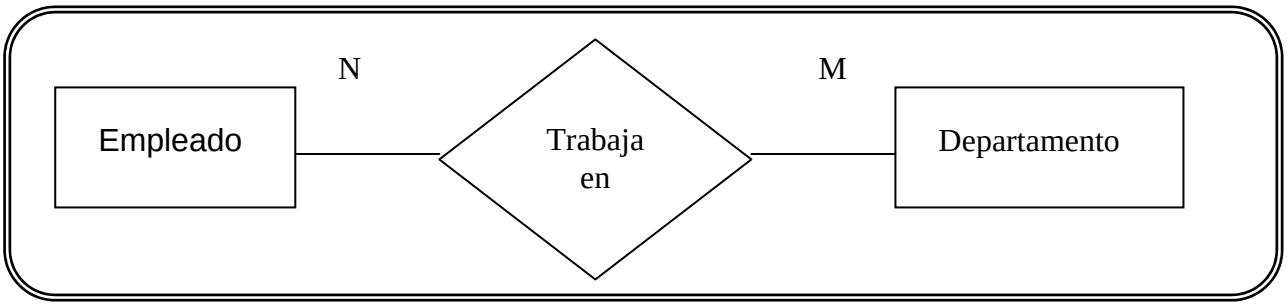
| | |
|---|---|
| Rectángulos: | Representan conjuntos de entidades. |
| Elipses: | Representan atributos. |
| Rombos: | Representan relaciones o vínculos. |
| Líneas: | unen atributos a conjuntos de entidades y conjuntos de entidades a conjuntos de relaciones. . |
| Elipses Dobles: | Representan atributos multivalorados. |
| Elipses Discontinuas: | Denotan atributos derivados. |

Líneas dobles: Indican participación total de una entidad en un conjunto de relaciones.

La Asociación se representa así:

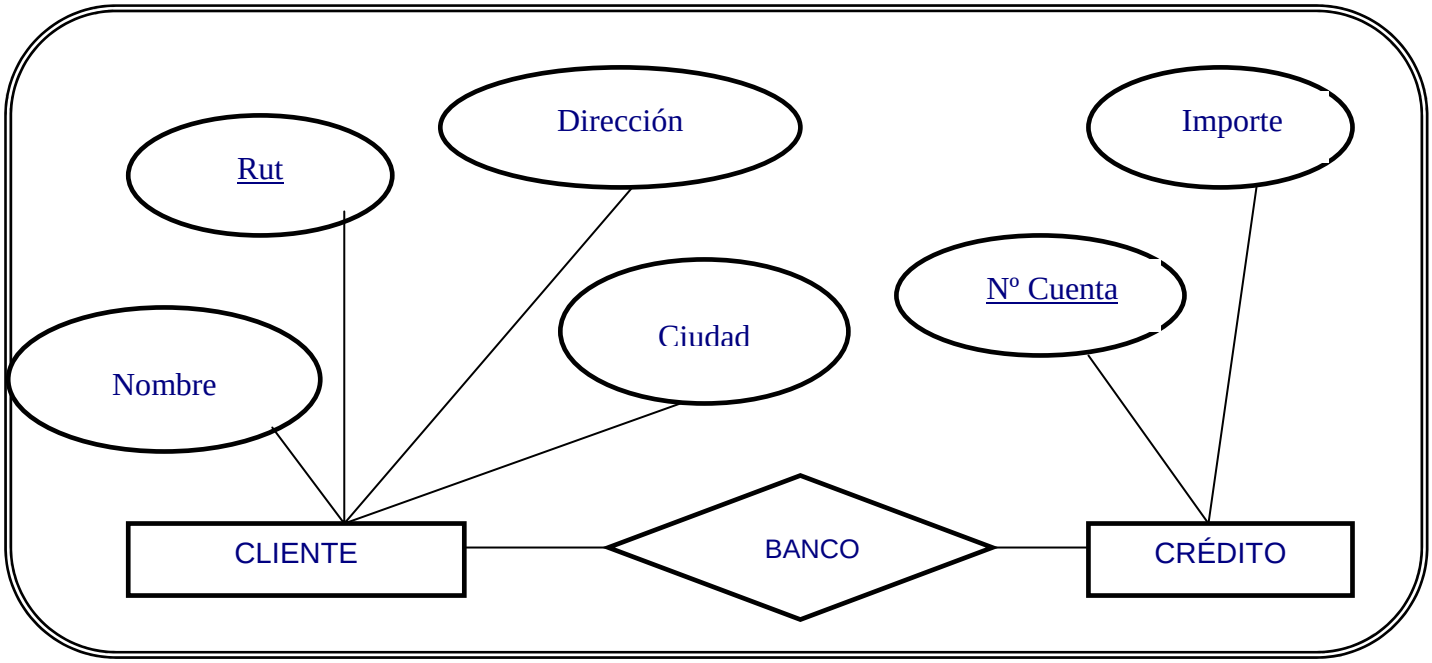


Ejemplo:




En el ejemplo anterior en el caso de empleados que trabajan en departamentos de una empresa, si bien un empleado en todo momento trabaja en un único departamento, puede hacerlo en distintos departamentos durante su permanencia en la empresa. Se utilizó una representación utilizando Diagrama E-R.

Ejemplo Nº 12



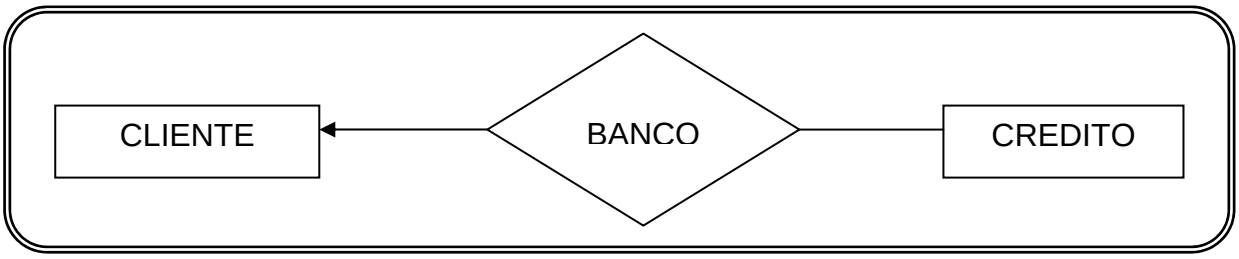
Como se muestra en el ejemplo, los atributos de un conjunto de entidades que son miembros de la clave primaria están subrayados para que el usuario se de cuenta que esos atributos son las llaves primarias para poder acceder a la información. Este diagrama E-R consta de dos conjuntos de entidades Cliente y Crédito. Los atributos asociados con cliente son: nombre, rut, dirección, y ciudad. Los atributos asociados con crédito son: nº de cuenta e importe.

No siempre es común la elección de los nombres para las entidades y los atributos. Debemos elegir nombres que comuniquen, hasta donde sea posible, los significados asignados a los distintos elementos de esquema. Para los tipos de entidad se usarán nombres en singular y con mayúscula. Los atributos comenzarán con mayúscula, así como también los vínculos.

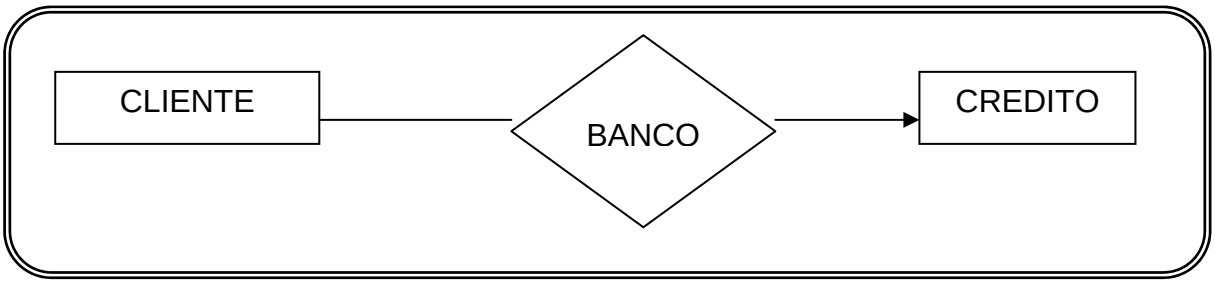
Realice ejercicios nº 11 al nº 15

El conjunto de relaciones BANCO puede ser varios es a varios, uno es a varios, varios a uno, o uno es a uno. Para distinguir entre estos tipos se dibuja una línea dirigida o una línea no dirigida ⟶ entre el conjunto de relaciones y las entidades en cuestión.

Volviendo al ejemplo Nº 12 del diagrama E-R, se ve que el conjunto de relaciones BANCO es varios a varios. Si el conjunto de relaciones BANCO fuera uno a varios, desde Crédito a Cliente, entonces la línea desde BANCO a Cliente sería dirigida, con una flecha apuntando al conjunto de entidades Cliente, como se muestra a continuación.




Ahora, si el conjunto de relaciones de BANCO fuera varios a uno, desde Cliente a Crédito, entonces la línea desde BANCO a crédito tendría una flecha apuntando al conjunto de entidades Crédito.




## CLASE 06

### 2.3. Conjuntos de Entidades Débiles.

Un conjunto de entidades puede no tener bastantes atributos para formar una clave primaria. Tal conjunto de entidades se denomina "conjunto de entidades débiles".

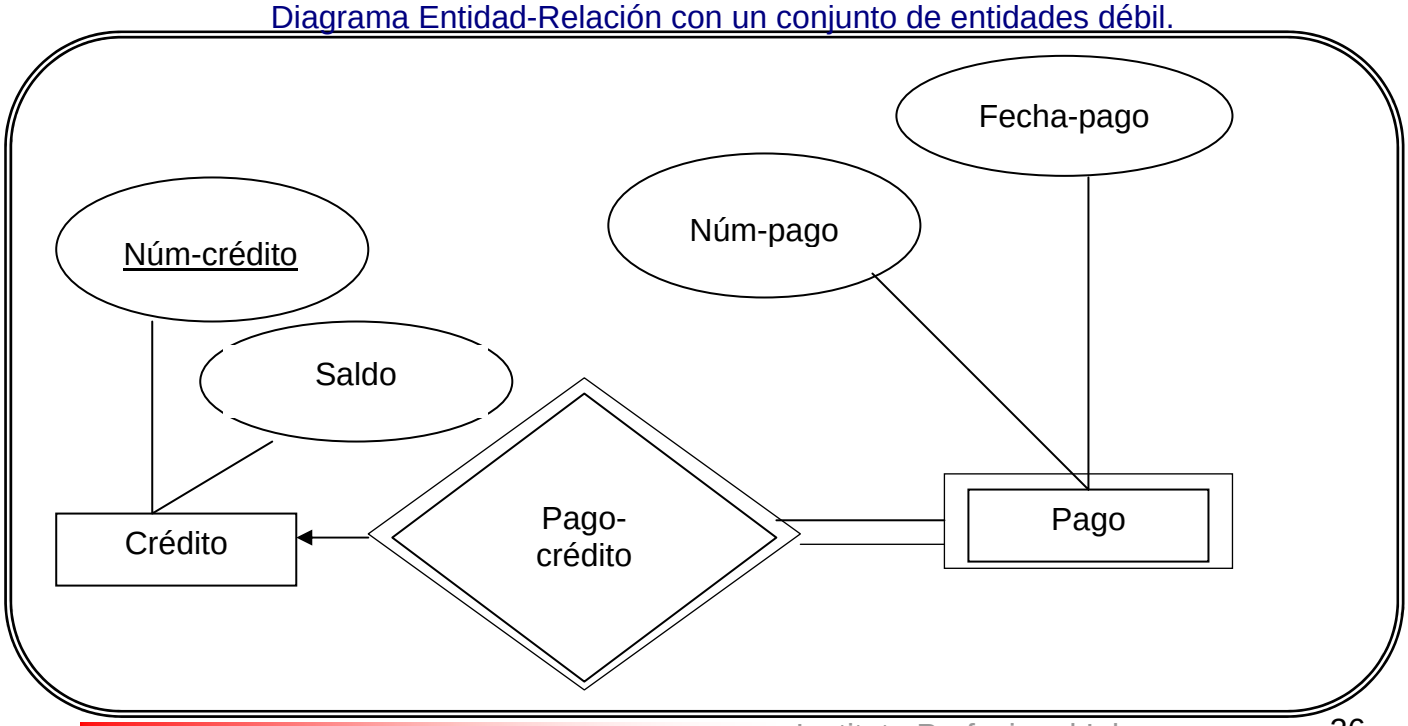
Aquellas entidades que poseen una clave primaria reciben el nombre de "conjunto de entidades fuertes".

Los conceptos de entidades fuertes y débiles están relacionados con las dependencias. Un miembro de un conjunto de entidades fuerte es por definición una entidad dominante, mientras que un miembro de un conjunto de entidades débil es una entidad subordinada.

El conjunto de entidades débil se indica en los diagramas E-R mediante un rectángulo dibujado con una línea doble y la correspondiente relación de identificación mediante un rombo dibujado con línea doble.

Ejemplo Nº 13

Diagrama Entidad-Relación con un conjunto de entidades débil.



En el ejemplo se puede observar, que el conjunto de entidades débil pago (para reconocerlo se encierra en un doble rectángulo) es dependiente del conjunto de entidades fuerte crédito a través del conjunto de relaciones pago-crédito. También en el ejemplo se observan líneas dobles para indicar participación total ( entidad pago con pago-crédito), lo que significa que cada pago debe estar relacionado a través de pago-crédito con alguna cuenta.

Finalmente la flecha pago-crédito a crédito indica que cada pago es para un único crédito.

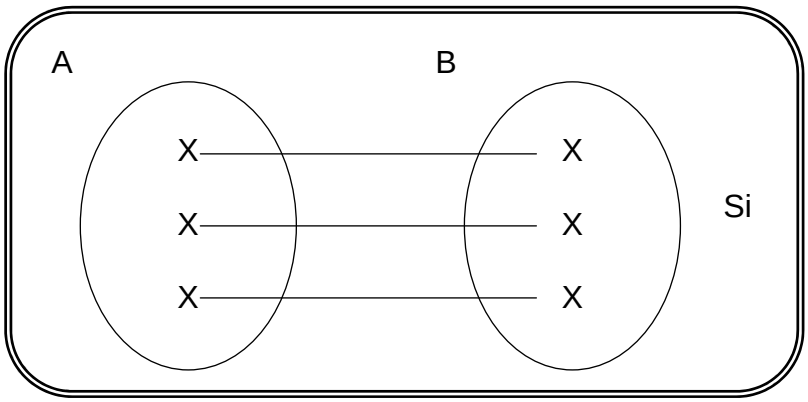
## 2.4. Dependencia Funcional.

La dependencia funcional es uno de los conceptos más importantes. Como definición podemos decir que es una restricción entre dos conjuntos de atributos de la base de datos.

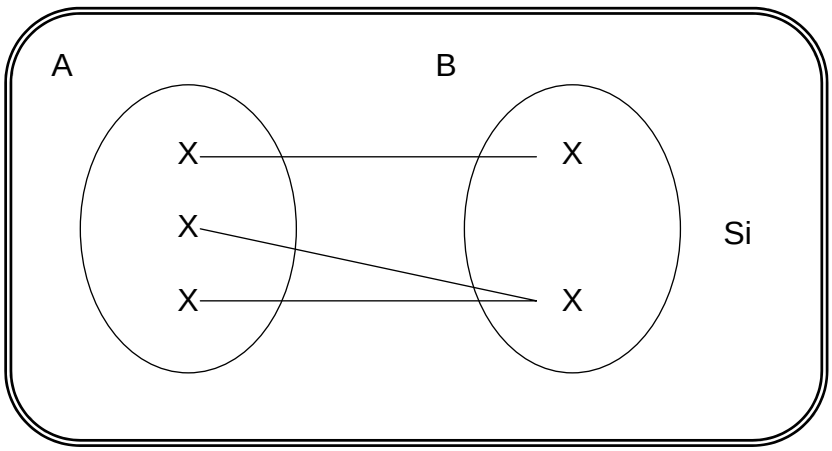
Se dice o se afirma que existe una dependencia funcional de B respecto de A, si para todo elemento de A hay un único elemento de B.

### Ejemplo N° 14

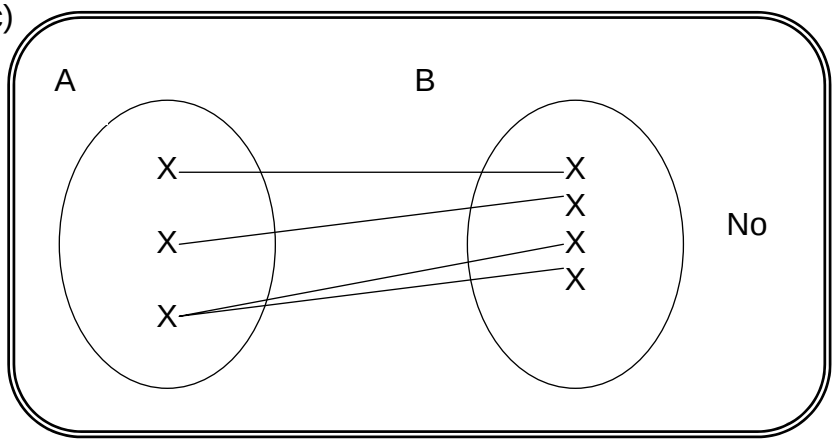
a)



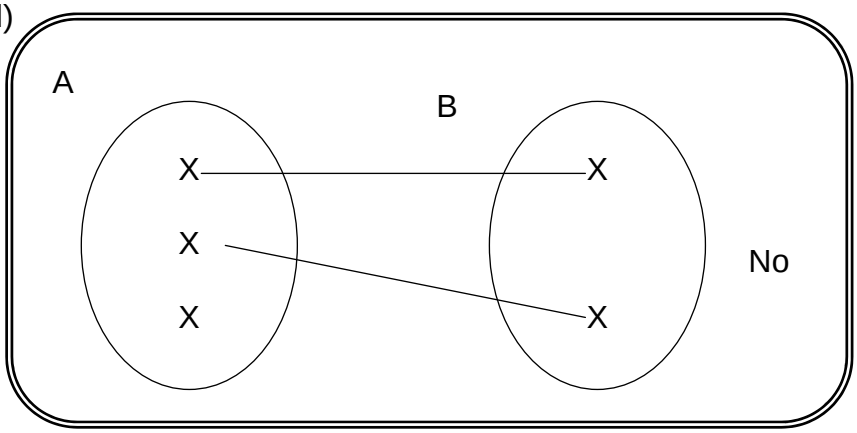

b)



c)




d)




Una dependencia funcional, denotada por X ⟶ Y, entre dos conjuntos X e Y, especifica una restricción que dice que dos tuplas[2] cualesquiera A1 y A2, de r tales que A1(X) = A2(X), pero también debemos tener a A1(Y) y A2(Y). Lo que quiere decir es que los valores del componente Y de la tupla r dependen de los valores del componente X, es decir, que los valores de X determinan funcionalmente los valores de Y.

Entonces si X determina funcionalmente a Y si y sólo si, siempre que dos tuplas en r(R) coincidan en su valor X, forzosamente deben coincidir en su valor Y. Podemos deducir que:

- Si una restricción de R que no puede haber más de una tupla con un valor X, es decir que X es una clave candidata de R, lo que significa que X ⟶ Y en R, para cualquier subconjunto de atributos Y de R.

- Si X ⟶ Y en R, pero no se refiere a que si Y ⟶ X o no.

[2] Tuplas: son los registros, las filas de una tabla que tienen la información de los atributos .

Por ejemplo el rut de una persona determina de manera única el nombre de la persona a quien le corresponde éste. Esto es una dependencia funcional pues condiciona que el rut corresponda a esa persona.

Supóngase que nos dan un esquema de relaciones R = (A, B, C, G, H, I) y el conjunto de dependencias funcionales:

a) $A \longrightarrow B$

b) $A \longrightarrow C$

c) $G \longrightarrow H$

d) $CG \longrightarrow I$

e) $B \longrightarrow H$

La dependencia funcional:

$A \longrightarrow H$ Se demuestra que si A pertenece a B y B pertenece H, entonces A pertenece a H

Se implica lógicamente. Es decir, podemos demostrar que siempre que se cumpla el conjunto dado de dependencias, $A \longrightarrow H$ también debe cumplirse.

Dado un conjunto de relaciones cada uno de los elementos del conjunto estará en cada elemento del mismo. Todos pertenecen a todos.

Las dependencias funcionales son propiedades de los significados de los atributos. Para poder especificar las dependencias funcionales los diseñadores de base de datos aprovechan su conocimiento en la relación de los atributos en un conjunto de relaciones.

Siempre que la semántica[3] de dos conjuntos de atributos de R indique que debe cumplirse una dependencia funcional, se especificará esa dependencia como una restricción.

Las dependencias funcionales se usan de dos formas, éstas son:

[3] Semántica[3], quiere decir el método que se utiliza para demostrar la condición que se quiere que se cumpla.

1. Para especificar restricciones, es decir, se interesa por las relaciones que satisfagan un conjunto dado de dependencias funcionales. Si queremos limitarnos a las relaciones de esquema R que satisfacen F, decimos que F se cumple en R.

   Si tenemos el siguiente conjunto:

   E:{ Rut ⟶ (Nombre, Dirección, Número-departamento),

   Numero-departamento ⟶ (Nombre-departamento, Rut-jefe) }

   Podemos decir que las dependencias funcionales serían:

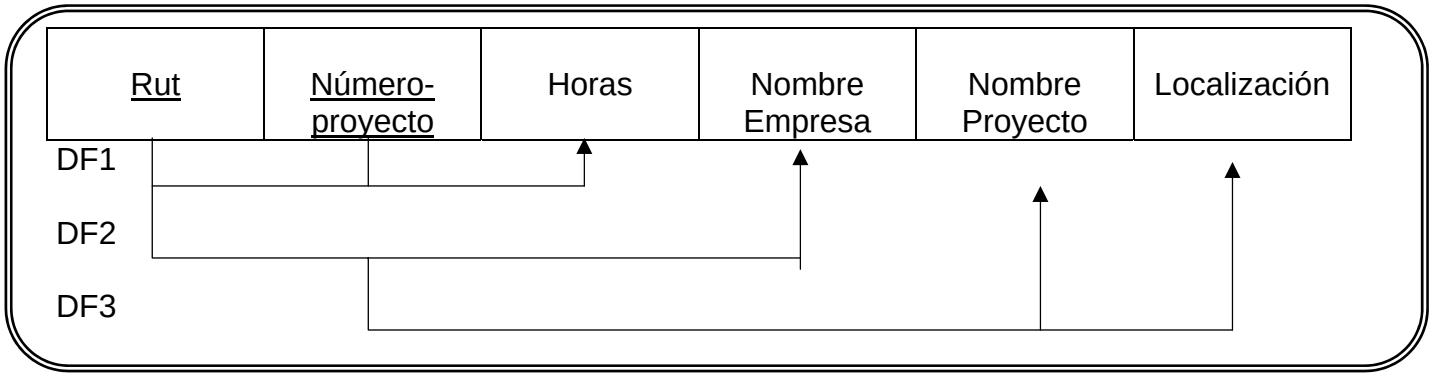
   Rut ⟶ {Nombre-departamento, Rut –jefe}

   Numero-departamento ⟶ Nombre-departamento

   Ya que Número-departamento está contenido en Rut, a Rut también le pertenece Nombre-departamento y Rut-jefe.

2. Para probar si una relación es legal bajo un conjunto dado de dependencias funcionales. Si una relación r es legal bajo un conjunto F de dependencias funcionales, decimos que r satisface a F.

Ejemplo:




En el ejemplo se puede observar que con la clave rut y número de proyecto se puede acceder a la información de la tupla. Por lo tanto hay tres dependencias funcionales como se ve. La primera nos indica que con rut y número de proyecto se puede acceder a las horas, la segunda dependencia es que con Rut se puede obtener el nombre de la empresa y la tercera que con el número de proyecto se puede acceder a nombre de proyecto y localización.

De Rut y número de proyecto dependen los demás subconjuntos, son dependencias funcionales.

Ejemplo de dependencia funcional.

Un atributo B, depende funcionalmente de otro atributo A de la misma entidad, si a cada valor de A, le corresponde sólo un valor de B. Ejemplo: en la entidad ALUMNO cuyos atributos son:

Número de Matricula (clave):

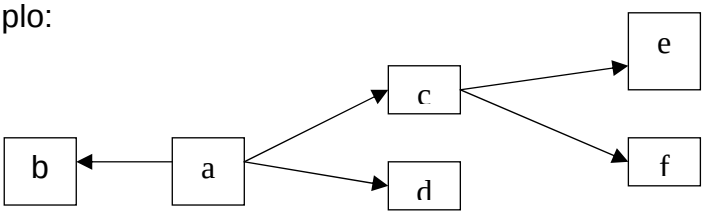
Nombre

Carrera

Dirección

Los atributos Nombre, Carrera y Dirección dependen funcionalmente de la clave Número de Matrícula, para acceder a los demás datos.

Dependencia funcional completa: un atributo B tiene dependencia funcional completa de un grupo de atributos A de la misma entidad, si B depende funcionalmente de A pero no de ningún subconjunto obtenido de los posibles atributos que forman A. Por ejemplo:




Dependencias funcionales: a $\longrightarrow$ b, c, d y c $\longrightarrow$ e, f

## CLASE 07

### 2.5. Dependencia transitiva

Sean A, B y C tres atributos (o grupos de atributos) de una relación. Si B depende funcionalmente de A y C de B, pero A no depende funcionalmente de B se dice que C depende transitivamente de A.

Existen seis reglas para trabajar las dependencias transitivas:

1. R1: es una regla reflexiva que nos indica que un conjunto de atributos siempre se determina a sí mismo. Crea dependencias que siempre son verdaderas.

Regla reflexiva: Si X pertenece al conjunto de Y, entonces X $\longrightarrow$ Y.

Ejemplo:

| Nº | Nombre |
|---|---|
| 1 | María |
| 2 | Juan |

X $\longrightarrow$ Y

Nº $\longrightarrow$ Nombre

2. R2: es una regla de aumento la cual señala que añadir el mismo conjunto de atributos a los dos miembros, tanto izquierdo o derecho, de una dependencia produce otra dependencia.

Regla de aumento: Si X $\longrightarrow$ Y = XZ $\longrightarrow$ YZ

Ejemplo:

Rut $\longrightarrow$ Nombre

Rut $\longrightarrow$ dirección $\longrightarrow$ teléfono

Si con el Rut se determina el nombre de una persona, entonces con el Rut más la dirección y el teléfono también se determina el nombre.

3. R3: las dependencias funcionales son transitivas.

Regla transitiva: Si X $\longrightarrow$ Y, Y $\longrightarrow$ Z = X $\longrightarrow$ Z

Ejemplo:

Rut $\longrightarrow$ Nombre

Rut $\longrightarrow$ dirección

Rut $\longrightarrow$ teléfono

Si con el Rut puede determinarse el código postal de la comuna de residencia de una persona y con este código puede determinarse el nombre de la comuna, entonces con el Rut puede determinarse el nombre de la comuna. Éste es el mecanismo básico de funcionamiento del enlace entre tablas a partir de claves

4. R4: es una regla que dice que se puede quitar los atributos del miembro derecho de una dependencia.

Regla de descomposición o proyectiva:

X $\longrightarrow$ YZ = X $\longrightarrow$ Y

Ejemplo:

Rut $\longrightarrow$ Nombre, dirección

Rut $\longrightarrow$ dirección

Si a partir del Rut es posible deducir el nombre y la dirección de una persona, entonces con el Rut también es posible determinar sólo la dirección.

5. R5: es una regla que nos permite hacer lo opuesto, combinar un conjunto de dependencias en una sola dependencia funcional.

Regla de unión o aditiva:

$$X \longrightarrow Y,\ X \longrightarrow Z = X \longrightarrow YZ$$

Ejemplo:

Rut ⟶ Nombre

Dirección ⟶ Teléfono

Rut, dirección ⟶ Nombre, teléfono

Si con el Rut se determina el nombre y con la dirección el teléfono de una persona, entonces con el Rut y la dirección podrá determinarse el nombre y el teléfono.

6. R6: rs una regla pseudotransitiva.

Regla de pseudotransitiva:

$$X \longrightarrow Y,\ WY \longrightarrow Z = WX \longrightarrow Z$$

Ejemplo:

Entidad EMPLEADO cuyos atributos son:

- Número de Empleado (clave)
- Nombre
- Categoría
- Dirección

Número de Empleado ⟶ Nombre

Nombre, Dirección ⟶ Categoría

Nombre, Número de Empleado ⟶ Categoría

Si con el número de empleado puede determinarse el nombre del empleado y con el nombre puede determinarse la dirección, por lo tanto con la dirección y el nombre

se puede determinar la categoría. También hay una unión de tablas a partir de la clave.

Según Armstrong[4] las reglas de inferencia o relación R1, R2, R3 son correctas y se cumplen, lo que quiere decir, que un conjunto dado de dependencias funcionales F sobre un esquema de relación con cualquiera de las reglas de dependencia funcional R1, R2, R3, se cumplirá en todos los estados de relación r de R.

El empleo repetitivo de las reglas hará que existan dependencias funcionales hasta que sea posible y no puedan producir más. Se producirá un cierre que se representa de esta manera F+, y esto se realizará solamente utilizando las reglas de inferencia de Armstrong.

Realice Ejercicios Nº 16 al Nº 20

## CLASE 08

### Axiomas Básicos:

- A1: Reflexividad

  Si $Y \rightarrow X$, $X \rightarrow Y$ (X , Y es una DF trivial)

- A2: Aumentatividad

  Si $X \rightarrow Y$ y $Z \rightarrow Y$, entonces $XY \rightarrow YZ$

- A3: Transitividad

  Si $X \rightarrow Y$ e $Y \rightarrow Z$, entonces $X \rightarrow Z$

### Axiomas Derivados:

- D1: Proyectividad

  Si $X \rightarrow Y$, entonces $X \rightarrow Y'$ si $Y' \rightarrow Y$

[4]W. Armstrong 1974, deduce las reglas de inferencia o axiomas que ayudan a que la relación cumpla con todos los estados de dependencia funcional que tenga. Permite entender de mejor manera las relaciones y su acceso.

- D2: Unión a aditividad

  Si $X \longrightarrow Y$ y $X \longrightarrow Z$, entonces $X \longrightarrow YZ$

- D3: Pseudotransitividad

  Si $X \longrightarrow Y$ e $YW \longrightarrow Z$, entonces $XW \longrightarrow Z$

Aunque los axiomas de Armstrong facilitan un procedimiento algorítmico para calcular el cierre DF+ de un conjunto de dependencias, su cálculo consume mucho tiempo, ya que, aunque el número inicial de dependencias sea pequeño, el número total de dependencias en el cierre es muy elevado.

Para evitar este problema habrá que buscar procedimientos algorítmicos que no estén basados en el cierre de un conjunto de dependencias.

## 2.6. Relaciones en que participan más de dos entidades.

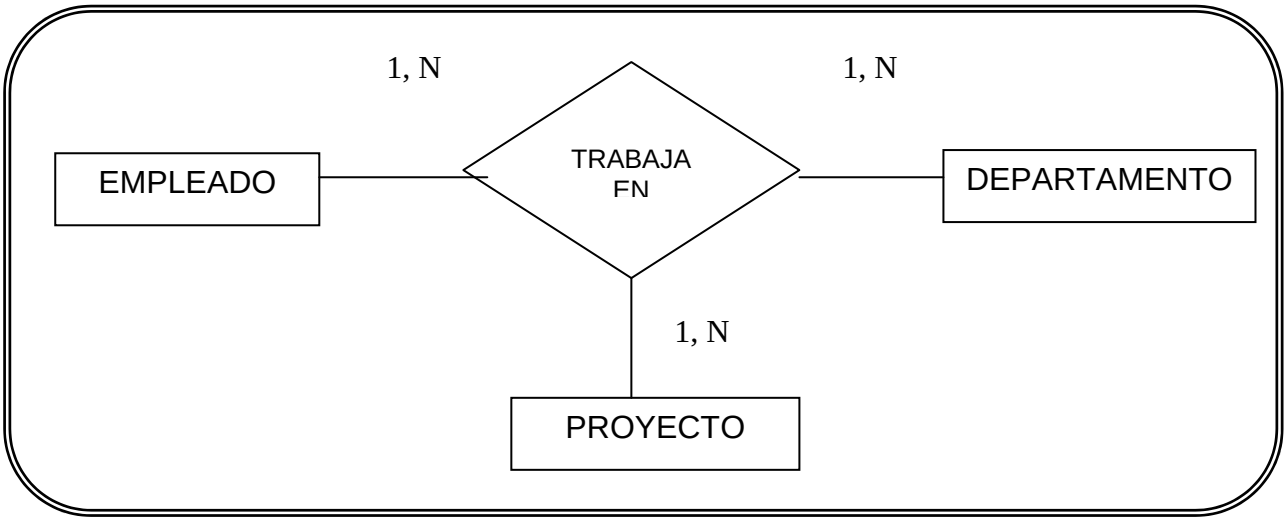
Este es uno de los casos interesantes, dado que por lo general se intenta tratar con relaciones de una dimensión (que están definidas como el nº de entidades que participan en la relación) o a lo sumo igual a 3.

Esto se explica por la dificultad que se presenta en términos de interpretación del significado de una relación, sobre todo si su dimensión es superior a 3. En este sentido se suele buscar descomponer las relaciones, entendiendo como tal el reemplazo de la relación por varias relaciones de dimensión inferior al de la relación original.

A continuación se verá el caso más frecuente:

Caso 1: un atributo o grupo de atributos de la relación no depende más de un subconjunto del identificador de la relación.

Ejemplo N° 15:




En que:

EMPLEADO (nº-empleado, nombre, dirección)

PROYECTO (nº-proyecto, nombre_proyecto)

DEPARTAMENTO (nº-depto, nombre_departamento)

TRABAJA_EN (EMPLEADO, PROYECTO, DEPARTAMENTO, fecha-inicial, fecha-final, tipo-trabajo)

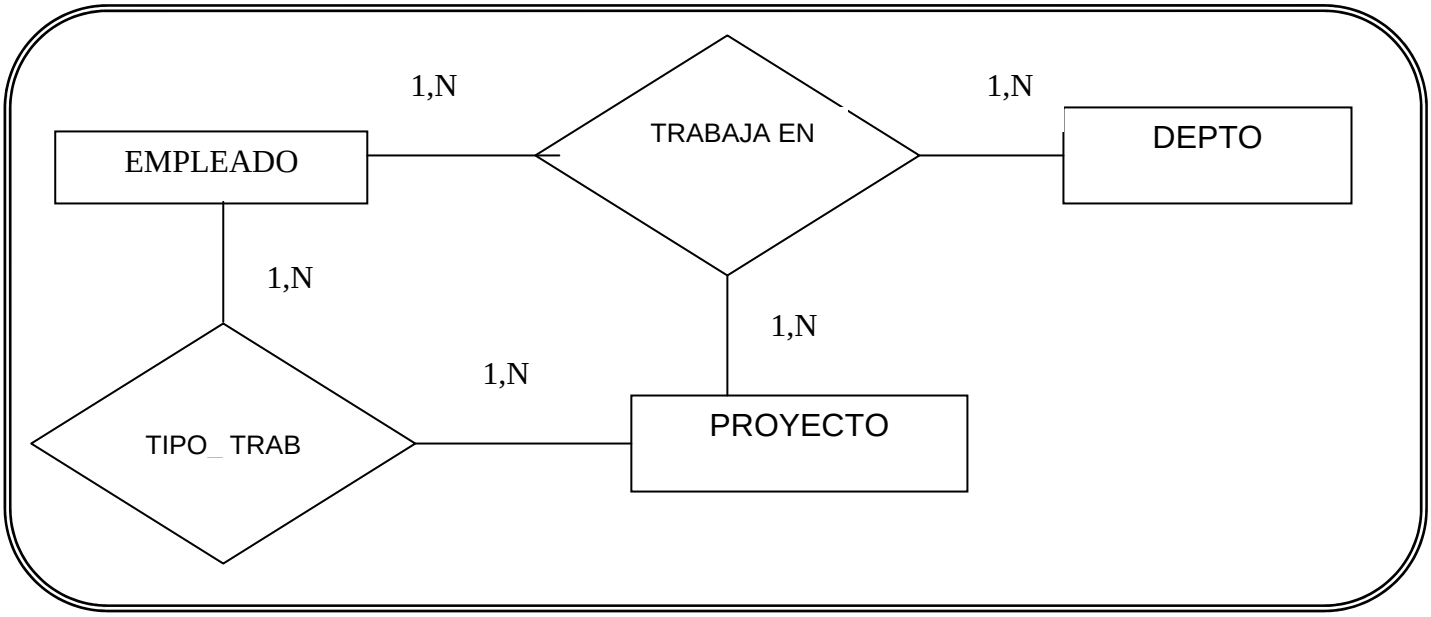
Si se toma en cuenta que existe una dependencia funcional del conjunto de atributos (nº_empleado, nº_proyecto) que dependen funcionalmente del atributo tipo_trabajo, esto es, una restricción entre los dos conjuntos de atributos de la base de datos. Se debe saber que tipo de trabajo es para saber el empleado y el proyecto. Sea R un esquema de relación que este último atributo no depende del departamento.

En este sentido podemos observar que la relación TRABAJA_EN explica:

a) El tiempo que es asignado a un empleado en un departamento para un proyecto dado.

b) El tipo de trabajo que un empleado presta de un proyecto.

Estas especificaciones se podría representar por dos relaciones, de la siguiente manera:




Donde las entidades conservan los atributos originales, en tanto que las relaciones quedan como a continuación se muestra:

TRABAJA_EN (EMPLEADO, DEPTO,PROYECTO, fecha-inicial, fecha-final,)

TIPO-TRABAJO (EMPLEADO, PROYECTO, tipo-trabajo)

Como se ha visto el modelo entidad-relación se apoya en el concepto de dependencia funcional, y a continuación el proceso de normalización que se estudia redefine de forma tal que las estructuras de las relaciones, sean las óptimas para el punto de vista lógico y se comprendan de mejor manera dichas estructuras.

## CLASE 09

### 2.7. Proceso de Normalización de una Relación

Las relaciones, tal como han sido definidas, pueden llegar a tener propiedades no deseables, como lo es la inconsistencia de la información contenida en las relaciones.

Por ejemplo que una misma persona aparezca en una misma relación o en más de una, con dos direcciones diferentes. Ello puede ocurrir cuando se actualizan relaciones, ya sea porque en una misma relación dicha persona aparece más de una vez, así como también porque dicha persona, con su dirección aparece en más de una relación y solo se actualiza una ocurrencia o tupla de la relación en el primer caso, o solo se actualiza la ocurrencia de una relación, pero no la de la otra relación en el segundo caso.

La normalización es un proceso paso a paso de análisis de los datos destinado a eliminar estas propiedades indeseables, y está basado en la Dependencia funcional (DF)[5] y la Dependencia Transitiva.

Según la propuesta de Codd[6] (1972), el proceso de normalización se somete a una serie de pruebas para certificar si pertenece o no a una cierta forma normal[7]. Este proceso sigue un estilo descendente[8] evaluando las relaciones a medida que sea necesario.

La normalización de los datos puede considerarse como un proceso de análisis de los esquemas de relación dados, basado en sus DF y claves primarias para alcanzar las propiedades deseables como lo son:

1.- Minimizar la redundancia

2.- Minimizar las anomalías de inserción, eliminación y actualización.

Los esquemas que no cumplan con las condiciones, pruebas de formas normales, se descomponen en esquemas de relación más pequeños que satisfagan las pruebas y de este modo posean las propiedades deseables.

El procedimiento de normalización proporciona a los diseñadores los siguientes aspectos:

- Un marco para analizar los esquemas basándose en sus claves y en las dependencias funcionales entre sus atributos.

- Una serie de pruebas de formas normales que pueden efectuarse sobre esquemas de relación individuales, de modo que la base de datos pueda normalizarse hasta el grado deseado.

Sea un ejemplo de las relaciones PROVEEDOR, PRODUCTO, PEDIDO con las siguientes afirmaciones:

PROVEEDOR (nº-proveedor, nombre-proveedor, ciudad)

[5] Siendo X e Y atributos de una sola relación, se afirma entonces que Y depende funcionalmente de X, si y solo si cada valor de X está asociado a un único valor de Y.

[6] En 1969 Edgar Codd inventó el modelo relacional, el modelo de bases de datos más usado hoy en día y para muchas personas.

[7] Forma normal se dice a la transformación de las relaciones para que queden en subconjuntos menores y se pueda actuar sobre ella sin perjudicar a las otras.

[8] Estilo descendente es descomponer la relación en un número de relaciones más pequeñas, de modo de optimizar el almacenamientos y sus funciones

PRODUCTO (nº_producto, nombre-producto, color)
PEDIDO (nº-proveedor, nº-producto, cantidad)

En esta relación PROVEEDOR se observa que los atributos nombre-proveedor, ciudad dependen funcionalmente de nº-proveedor. Es decir para un nº-proveedor dado, existe un único valor para cada uno de los dos atributos "dependientes". Decir que el atributo ciudad tiene DF de nº-provedor, equivale a afirmar que un proveedor debe estar necesariamente localizado en una única ciudad.

- Primera forma normal (1FN):

Se dirá que una relación se encuentra en primera forma normal si y solo si existe una DF de cada uno de los atributos que la conforman respecto del atributo identificador principal.

Se definió para prohibir los atributos multivaluados[9], los atributos compuestos y sus combinaciones. Establece que el dominio del atributo debe incluir sólo valores simples y que el valor de cualquier atributo en una tupla debe ser un valor individual proveniente del atributo de ese dominio.

La primera forma normal prohíbe las relaciones dentro de las relaciones, los únicos valores que permite son valores indivisibles, es decir, monovaluados.

Ejemplo N° 16:

Sea la relación PERSONA (rut, nombre, dirección, fono). Se sabe que toda persona tiene una única dirección, pero puede no tener asignado fono alguno, así como puede tener más de un fono asignado a su nombre, cabe agregar que todo fono sólo puede estar asignado a una persona.

En este ejemplo la relación PERSONA no está en 1FN porque no existe un Dependencia funcional del atributo fono respecto de la clave rut.

Observemos las siguientes anomalías en la tabla:

| Rut | Nombre | Dirección | Fono |
|---|---|---|---|
| 034 | Raúl | 3 oriente 2064 | 255696 |
| 025 | Juan | 18 poniente 26 | 365956 |
| 014 | Juan | 18 poniente 26 | 295752 |
| 030 | María | 15 norte 465 | ---------- |
| 020 | Claudio | Alameda 1164 | 256978 |

[9] Multivaluados son atributos que pueden tener un conjunto de valores para la misma entidad

Raúl y Claudio tienen un fono cada uno, mientras que Juan tiene dos y María no tiene. Si Raul contratara una nueva línea con fono 287956, corresponde incorporar una nueva tupla.

Existe la posibilidad que la tabla quede de la siguiente manera:

| Rut | Nombre | Dirección | Fono |
|---|---|---|---|
| 034 | Raúl | 3 oriente 2064 | 255696 |
| 034 | Raúl | 4 oriente 2064 | 287956 |
| 025 | Juan | 18 poniente 26 | 365956 |
| 025 | Juan | 18 poniente 26 | 295752 |
| 030 | María | 15 norte 465 | ---------- |
| 020 | Claudio | Alameda 1164 | 256978 |

Vea que se incorporó una nueva tupla en Raúl en la tabla con un "error" en la dirección, en lo que se conoce como error de inserción.

Realice ejercicios nº 21 al nº 25

También rut ya no sería clave de la relación porque no identifica las tuplas al haber más de una misma tupla con la misma clave. Por lo tanto la clave debería ser rut más fono en unión.

En consecuencia el primer paso en un proceso de normalización es remover aquellos grupos de atributos que no tengan dependencias funcionales con respecto a la clave de la relación repetitivos. Cada uno de estos grupos redefine una nueva relación en la que se incluya la clave de la relación original como clave foránea, que será parte de la clave de la nueva relación. La tabla quedaría:

| Rut | Nombre | Dirección | Fono | Rut |
|---|---|---|---|---|
| 034 | Raúl | 4 oriente 2064 | 287956 | 034 |
| 025 | Juan | 18 poniente 26 | 365956 | 025 |
| 025 | Juan | 18 poniente 26 | 295752 | 025 |
| 030 | María | 15 norte 465 | ---------- | 030 |
| 020 | Claudio | Alameda 1164 | 256978 | 020 |

Por lo tanto una relación sin grupos de atributos repetitivos se encuentra en 1 FN.

## CLASE 10

- Segunda forma normal (2FN):

Se dirá que una relación se encuentra en su 2FN cuando en su 1FN cada atributo que no forma parte de la clave tiene dependencias funcionales completas respecto de la clave.

Una dependencia funcional X $\longrightarrow$ Y es una dependencia funcional completa, si la eliminación de cualquier atributo A de X hace que la dependencia deje de ser válida, vale decir, para cualquier atributo A pertenece a X, (X menos A) no determina funcionalmente a Y.

Una dependencia funcional X $\longrightarrow$ Y es una dependencia parcial si es posible eliminar un atributo A que pertenece a X de X y la dependencia sigue siendo válida. Vale decir, para algún A pertenece a X, (X menos A) igual depende de Y.

### Ejemplo N° 17:

Sea la relación: COMPOSICIÓN (nº-producto, nº-pieza, rut-proveedor, ciudad, cantidad).

Se sabe que cada pieza es aprovisionada por un único proveedor que tiene permanencia en una única ciudad. Veamos la siguiente tabla:

| Nº-producto | Nº -pieza | Rut-proveedor | Ciudad | Cantidad |
|---|---|---|---|---|
| S1 | P1 | 1 | Linares | 300 |
| S1 | P2 | 2 | Talca | 200 |
| S1 | P3 | 2 | Talca | 400 |
| S1 | P4 | 2 | Talca | 200 |
| S1 | P5 | 3 | Rancagua | 100 |
| S1 | P6 | 2 | Talca | 100 |
| S2 | P1 | 1 | Linares | 300 |
| S2 | P2 | 2 | Talca | 400 |
| S3 | P2 | 2 | Talca | 200 |
| S4 | P2 | 2 | Talca | 200 |
| S4 | P4 | 2 | Talca | 300 |
| S5 | P5 | 3 | Rancagua | 400 |

Observar que esta relación puede presentar problemas a la consistencia de la información cuando se desee realizar cualquiera de las siguientes operaciones:

a) Inserción: impide incluir una nueva pieza en tanto se desconozca el producto que lo contiene, porque el atributo nº-producto no puede asumir un valor nulo dado que forma parte de la clave.

b) Eliminación: al eliminar una composición se eliminan también los datos relativos a la pieza asociada al producto. Si se elimina la sexta ocurrencia con nº-producto = S1 y nº-pieza =P6 se está perdiendo la información de la pieza P6.

c) Actualización: si el proveedor de la pieza P2 se traslada de Talca a Rancagua se corre el riesgo de actualizar la segunda ocurrencia y dejar sin actualizar otras ocurrencias en que esté involucrado el mismo proveedor.

Los problemas mencionados aparecen porque la relación no está en 2 FN. Ello ocurre porque la relación COMPOSICIÓN no tiene dependencia funcional completa de los atributos rut-proveedor y ciudad respecto de la clave, dado que existe dependencia funcional con un subconjunto de la clave (nº-pieza).

Para resolver estas anomalías detectadas es necesario reformular la relación COMPOSICIÓN, la cual quedará de la siguiente forma:

COMPOSICIÓN: (nº-producto, nº pieza, cantidad)

PIEZA: (nº-pieza, rut-proveedor, ciudad)

Representadas en dos Tablas quedará:

TABLA PIEZA

| Nº-pieza | Rut-prov | ciudad |
|---|---|---|
| P1 | 2 | Talca |
| P2 | 1 | Linares |
| P3 | 1 | Linares |
| P4 | 3 | Rancagua |
| P5 | 2 | Talca |

TABLA COMPOSICIÓN

| Nº-producto | Nº-pieza | Cantidad |
|---|---|---|
| S1 | P1 | 300 |

| | | |
|---|---|---|
| S1 | P2 | 200 |
| S1 | P3 | 400 |
| S1 | P4 | 200 |
| S1 | P5 | 100 |
| S1 | P6 | 100 |
| S2 | P1 | 300 |
| S2 | P2 | 400 |
| S3 | P2 | 200 |
| S4 | P2 | 200 |
| S4 | P4 | 300 |
| S5 | P5 | 400 |

La reestructuración de una relación a dos relaciones soluciona los problemas detectados:

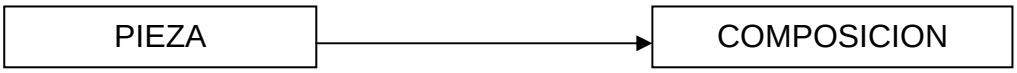
Inserción: ahora se puede incorporar una nueva pieza aún cuando no forme parte de ningún producto. Esto es, se puede incorporar una pieza en la relación PIEZA aún cuando no esté en COMPOSICIÓN.

Eliminación: es posible eliminar una composición sin que por ello se pierda información respecto de la pieza involucrada. En el ejemplo al eliminar el producto S3 no se está perdiendo la información de la pieza P2, puesto que ésta se encuentra en Tabla Pieza.

Actualización: se ha evitado la redundancia del atributo ciudad, bastando cambiar la única ocurrencia en la relación PIEZA cuando proveedor 2 se traslada de Talca a Rancagua.

Para obtener estos efectos positivos se eliminó el error que representaba el hecho de que en la primera versión de la relación COMPOSICIÓN se estaba en presencia de una dependencia funcional completa.

En el ejemplo el diagrama relacional es el siguiente:



Observación:

Si una relación está en la 1 FN, pero no en la 2 FN, es porque su clave es compuesta y no existe dependencia funcional completa de al menos uno de los atributos que componen la relación y que no forman parte de la clave.

Cabe destacar que si está en presencia de una relación cuya clave no es compuesta y que está en la 1 FN, entonces también estará en 2FN.

- Tercera forma normal (3FN):

Una relación está en tercera forma normal si todos los atributos de la relación dependen funcionalmente sólo de la clave, y no de ningún otro atributo. Es decir, una relación se encuentra en 3 FN cuando está en 2 FN y no existe dependencia transitiva entre los atributos.

Una dependencia funcional $X \longrightarrow Y$ en un esquema de R es una dependencia transitiva si existe un conjunto de atributos Z que no sea un subconjunto de cualquier clave de R y se cumple tanto $X \longrightarrow Y$ como $Z \longrightarrow Y$.

Ejemplo:

Siguiendo con el ejemplo de las relaciones COMPOSICIÓN Y PIEZA. Esta última sigue presentando dificultades originadas por la existencia de una dependencia funcional entre los atributos ciudad y rut-proveedor, ya que todo proveedor está en una única ciudad.

A continuación, se analizará que problema produce.

a) Inserción: impide incluir una nueva tupla (rut-proveedor, ciudad) si en la ciudad no se localiza ningún proveedor, ya que el atributo nº-proveedor. no puede asumir un valor nulo dado que es la clave de una relación PROVEEDOR.

b) Eliminación: si en la relación PROVEEDOR se elimina el único proveedor de una ciudad en particular, se perderá la información asociada a dicha ciudad, es decir, en la región en que se encuentra.

c) Actualización: una misma ciudad puede aparecer en varios registros, forzando a que en ellos aparezca el nombre de la misma región, con riesgo de incurrir en inconsistencias, como que una misma ciudad pudiese aparecer en distintas regiones.

Para resolver estos problemas se debe volver a descomponer la relación. En este caso, la relación PROVEEDOR se reestructura en las relaciones: PROVEEDOR Y ZONA.

Las tablas quedarían de la siguiente forma:

TABLA COMPOSICIÓN:

| Nº-prod | Nº -pieza | Cantidad |
|---|---|---|
| S1 | P1 | 300 |
| S1 | P2 | 200 |
| S1 | P3 | 400 |
| S1 | P4 | 200 |
| S1 | P5 | 100 |
| S1 | P6 | 100 |
| S2 | P1 | 300 |
| S2 | P2 | 400 |
| S3 | P2 | 200 |
| S4 | P2 | 200 |
| S4 | P4 | 300 |
| S5 | P5 | 400 |

TABLA PIEZA:

| Nº - pieza | Rut-proveedor |
|---|---|
| P1 | 2 |
| P2 | 1 |
| P3 | 1 |
| P4 | 3 |
| P5 | 2 |
| P6 | 2 |

TABLA PROVEEDOR:

| Rut-proveedor | Ciudad |
|---|---|
| 1 | Talca |
| 2 | Linares |
| 3 | Rancagua |

De esta forma se han eliminado los problemas de inserción, eliminación y actualización detectados en su momento. Y ahora lo único que se ha efectuado es eliminar la dependencia transitiva detectada en la relación PIEZA entre atributos que no formaban parte de la clave (rut-proveedor y ciudad).

Ahora las tres relaciones COMPOSICIÓN, PROVEEDOR Y PIEZA están en 3 FN y se asocian entre sí como lo muestra el diagrama relación en la siguiente representación:


# CLASE 11

## 3. CARACTERÍSTICAS DEL MODELO E-R

Los conceptos básicos de E-R pueden modelar la mayoría de las características de las bases de datos, pero existen aspectos que pueden expresar mediante extensiones del modelo E-R una mejora del modelo a través de la Especialización, la generalización, atributos heredados y la agregación.

### 3.1. Especialización.

La especialización es el proceso de definir un conjunto de subclases de un tipo de entidad.

Subclases se define como el conjunto de atributos de una entidad o conjunto de entidades.

El conjunto de subclases que forman la especialización se define a partir de alguna característica distintiva de las entidades.

Por ejemplo, Mujer está contenida dentro de persona, es decir, toda mujer es una instancia de la entidad persona, es una subclase de la entidad persona. Persona vendría siendo una superclase. Entonces mujer es una especialización de la entidad persona.

Otro ejemplo de Especialización es el siguiente, considere el conjunto de entidades Empleado con los atributos nombre-empleado, cargo-empleado. Pero además, la entidad Empleado se puede clasificar en empleado-asalariado y empleado por horas. Por lo tanto, existen dos especializaciones para la entidad Empleado.

El proceso de designación de subgrupos dentro de un conjunto de entidades es la Especialización.

Ejemplo, Empleado se divide en subgrupos de empleado-asalariado y empleado por horas. Por lo tanto, la especialización de la entidad Empleado sería empleado-asalariado y empleado por horas

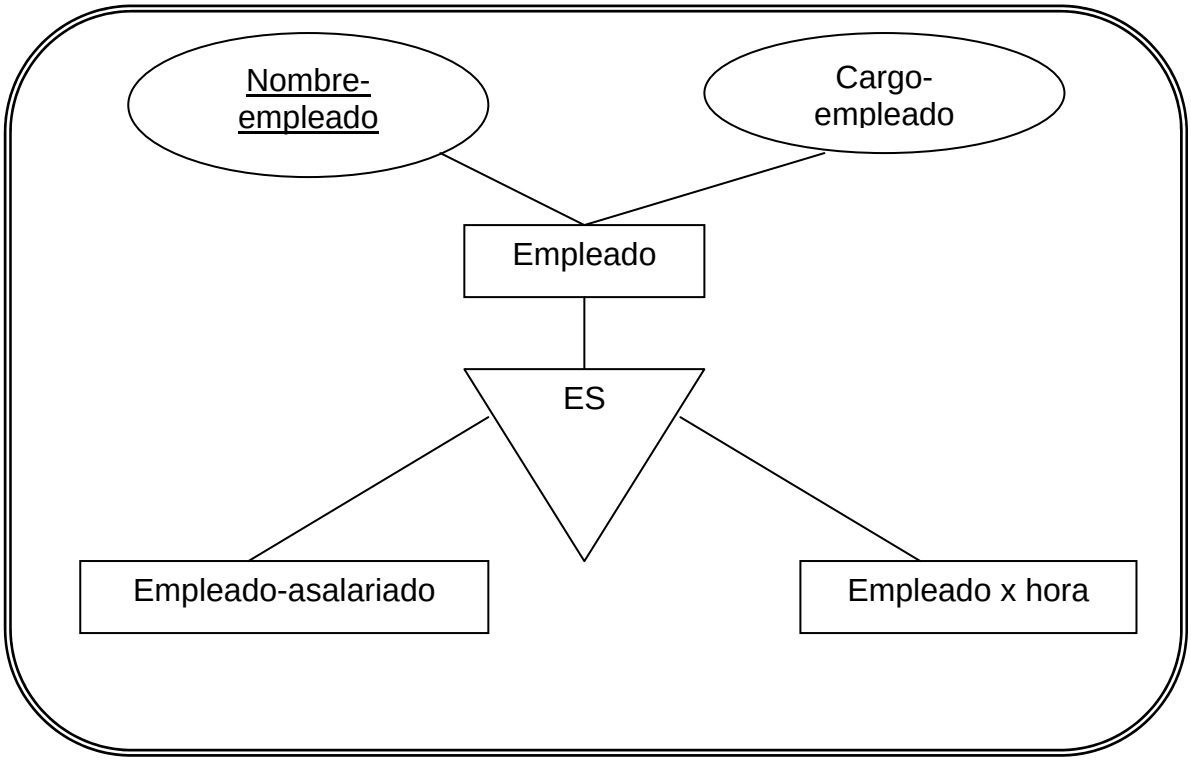
Se puede aplicar especialización repetidamente para redefinir el esquema de diseño.

Para simbolizar en el diagrama Entidad-Relación, la especialización se representa mediante un componente triangular etiquetado ES. La etiqueta ES representa que:

Un empleado-asalariado ES un empleado.

La relación ES se puede llamar también superclase-relación de subclase.

Ejemplo Nº 18:




En el ejemplo se representa que la entidad Empleado posee dos especializaciones que se dividen en los atributos nombre-empleado y Cargo-empleado y mediante la notación ES nos indica que la entidad Empleado – asalariado y Empleado por hora también son entidades que pertenecen a Empleado y que nacieron mediante la especialización. A su vez estas entidades tendrán nuevos atributos.

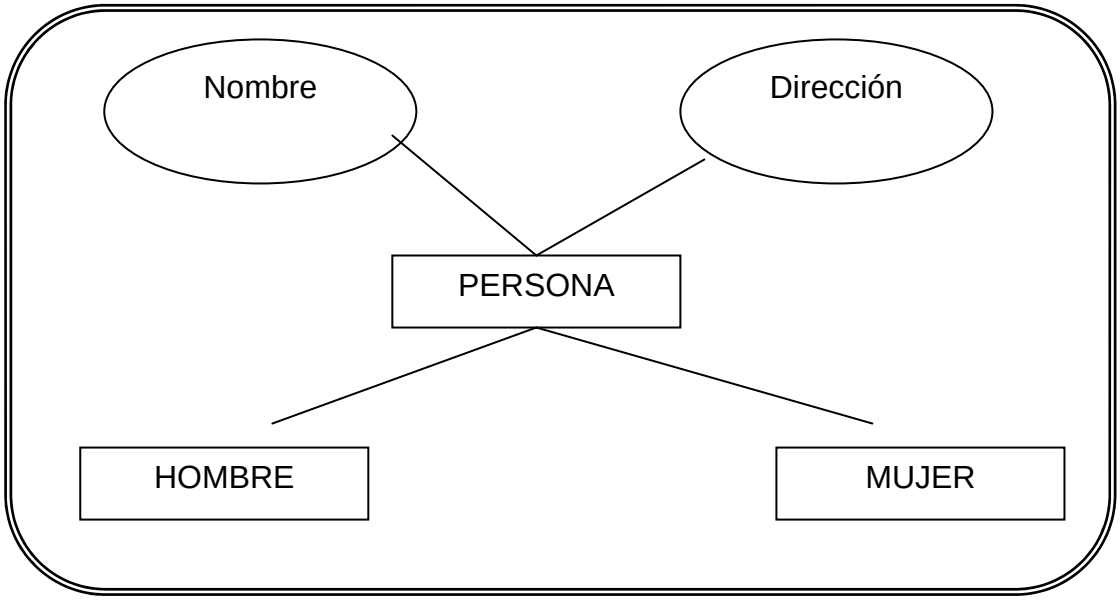
Cada vez que se divide una entidad en atributos se produce el fenómeno de especialización.

## 3.2. Generalización

Cuando un conjunto de entidades inicial se desglosa en sucesivos niveles de subgrupos de entidades representa un proceso de diseño de tipo Top-down. Este proceso también puede ser bottom-up o ascendente en donde los múltiples conjuntos de entidades se sintetizan en un conjunto de entidades de nivel más alto basado en características comunes.

Se denomina generalización al proceso de definir un tipo de entidad generalizado a partir de los tipos de entidad dados. Es una inversión simple de la especialización. Ambos se aplicarán en combinación en el curso de diseño del esquema E-R para un desarrollo.

Ejemplo Nº 19




En el ejemplo se puede observar que los tipos de entidad HOMBRE y Mujer, se pueden generalizar en el tipo de entidad PERSONA. Tanto Hombre como Mujer son subclases de la superclase generalizada PERSONA. Las flechitas indican para donde va la generalización.

Podemos observar que Hombre y Mujer son una especialización de PERSONA, antes que ver que PERSONA es una generalización de Hombre y Mujer

La generalización proviene del reconocimiento de un número de conjuntos de entidades que comparten algunas características. Fundadas en sus similitudes, la generalización sintetiza estos conjuntos de entidades en uno solo, el conjunto de nivel más alto.

### 3.3. Restricciones y Características de Especialización y Generalización.

Generalmente se pueden definir varias especializaciones o subclases para una misma entidad. Pero también existe la posibilidad de que una especialización pueda contener una subclase solamente.

Para algunas especializaciones se pueden determinar las entidades que se convertirán en miembro de cada subclase, especificando una condición en términos del valor de algún atributo de la superclase o entidad. Éstas reciben el nombre de subclases definidas por condición.

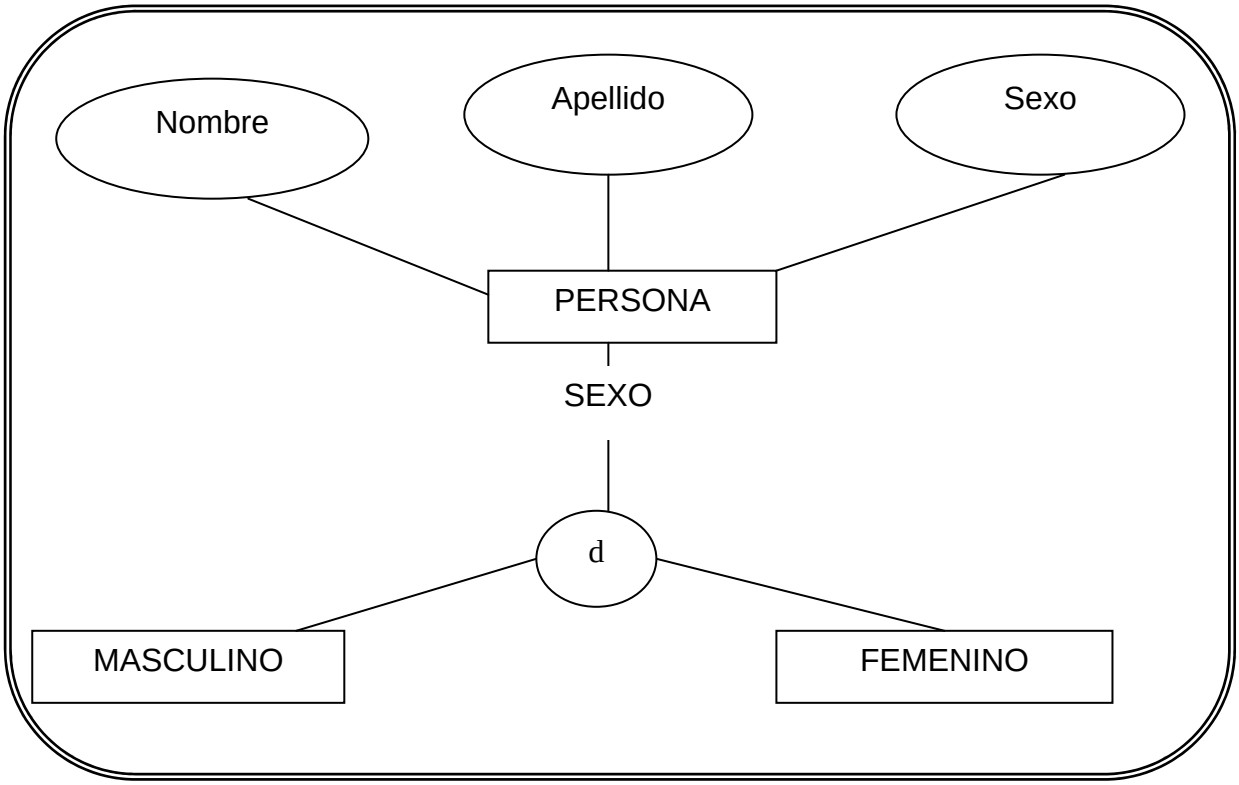
Por ejemplo, si se tiene una entidad Persona y ésta tiene un atributo llamado sexo, podemos especificar la condición de pertenencia donde sexo = Femenino. Esta condición es una restricción que especifica que los miembros de la subclase femenino deben satisfacer el predicado que es el que expresa un acontecimiento, el cual puede ser un estado de la condición que se debe cumplir y que todas las entidades del tipo Persona cuyo valor para el atributo sexo = femenino.

Cuando una especialización define una condición de pertenencia recibe el nombre de Especialización definida por atributo. Y cuando no hay una condición que determine la pertenencia a una entidad o subclase, se dice que está Definida por el usuario.

La pertenencia para cualquier subclase la define el usuario de la base de datos, cuando aplican la operación de añadir una entidad, así es el usuario el que especifica individualmente la pertenencia para cada entidad y no cualquier condición que pueda ser evaluado en forma mecánica.

También se pueden aplicar otras restricciones a una especificación. La primera restricción es la restricción de disyunción, que quiere decir que una entidad puede ser miembro de a lo menos una de las subclases de la especialización.

Ejemplo Nº 20:

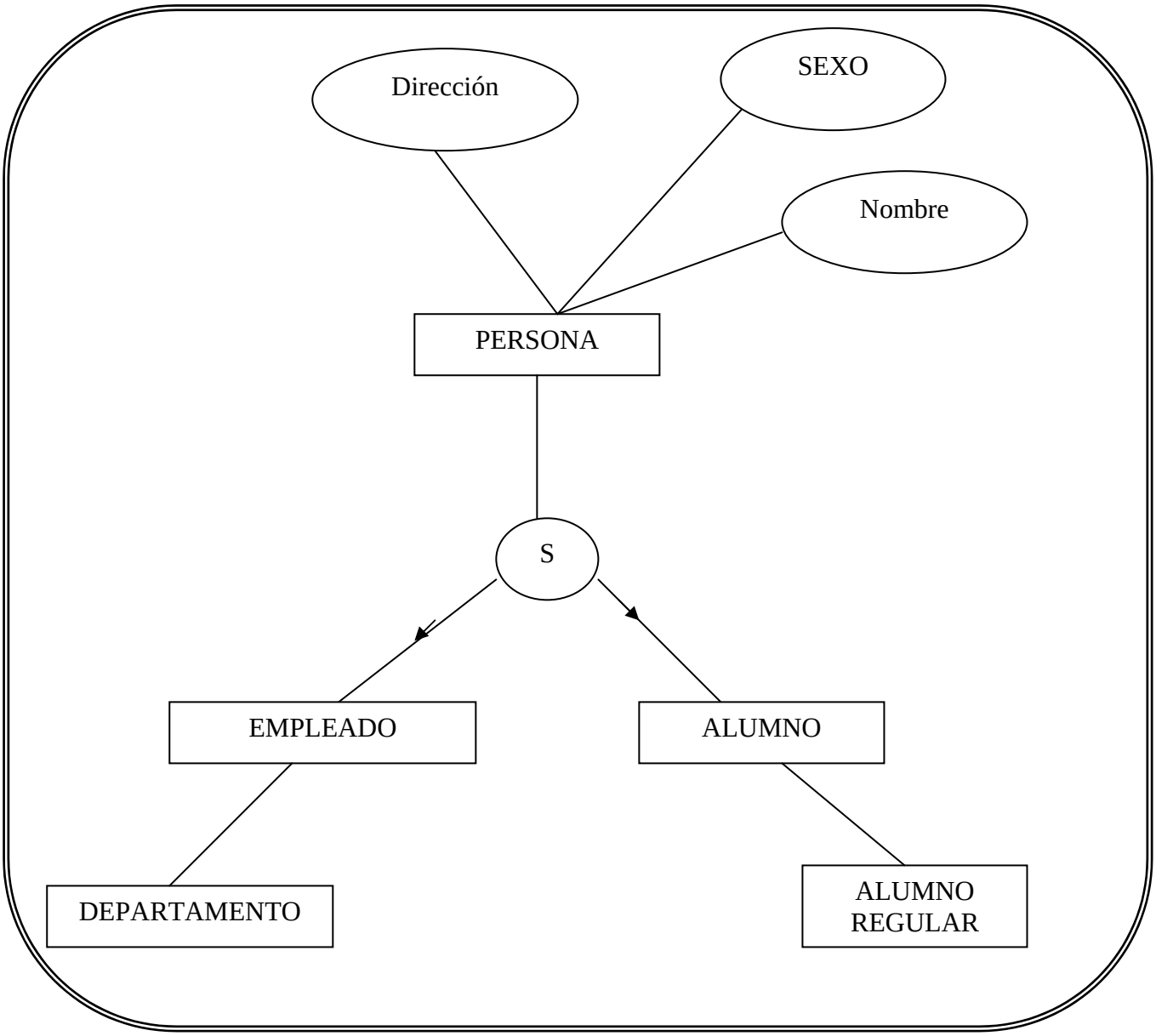



En el ejemplo se puede observar que el carácter disjunto (d) de conjuntos de entidades requiere que una entidad no pertenezca a más de un conjunto de entidades de nivel más bajo. La entidad PERSONA puede satisfacer sólo una condición para el atributo SEXO, una entidad puede ser bien MASCULINO o bien FEMENINO, pero no ambas cosas a la vez

También el conjunto de entidades puede ser Solapadas, la misma entidad puede pertenecer a más de un conjunto de entidades de nivel más bajo en una generalización simple.

# CLASE 12

## Ejemplo Nº 21:




En el ejemplo se observa como se produce una restricción con entidades solapadas, la misma entidad en este caso EMPLEADOS, puede ser miembro de más de una entidad de la especialización. Para indicar que es una entidad solapada se encierra en un circulo la letra S.

Una restricción final es la ligadura de completitud, en una generalización. Ésta especifica si un conjunto de entidades de nivel más alto debe pertenecer o no al menos a uno de los conjuntos de entidades de nivel más bajo en una generalización, esta ligadura puede ser:

a) Total: cada entidad de nivel más alto debe pertenecer a un conjunto de entidades más bajo.

b) Parcial: algunas entidades de nivel más alto pueden no pertenecer a algún conjunto de entidades de nivel más bajo.

Ejemplo Nº 22:

La generalización de Empleado es total, pues todos los empleados deben ser Empleados asalariados o bien Empleados por horas. Debido a que el conjunto de entidades más alto alcanzado a través de la generalización está generalmente compuesta solamente por esas entidades del conjunto de nivel más bajo (empleado –asalariado y empleados por horas) entonces la ligadura para este tipo de entidad es total.

## 3.4. Agregación

La agregación consiste en interpretar la idea "compuesto de", identifica entidades que participan en otra entidad y atributos que en sí están compuestos por otros atributos, es decir, existen casos donde será necesario agrupar dos o más conjuntos de entidades relacionados para conformar un solo conjunto de entidades.

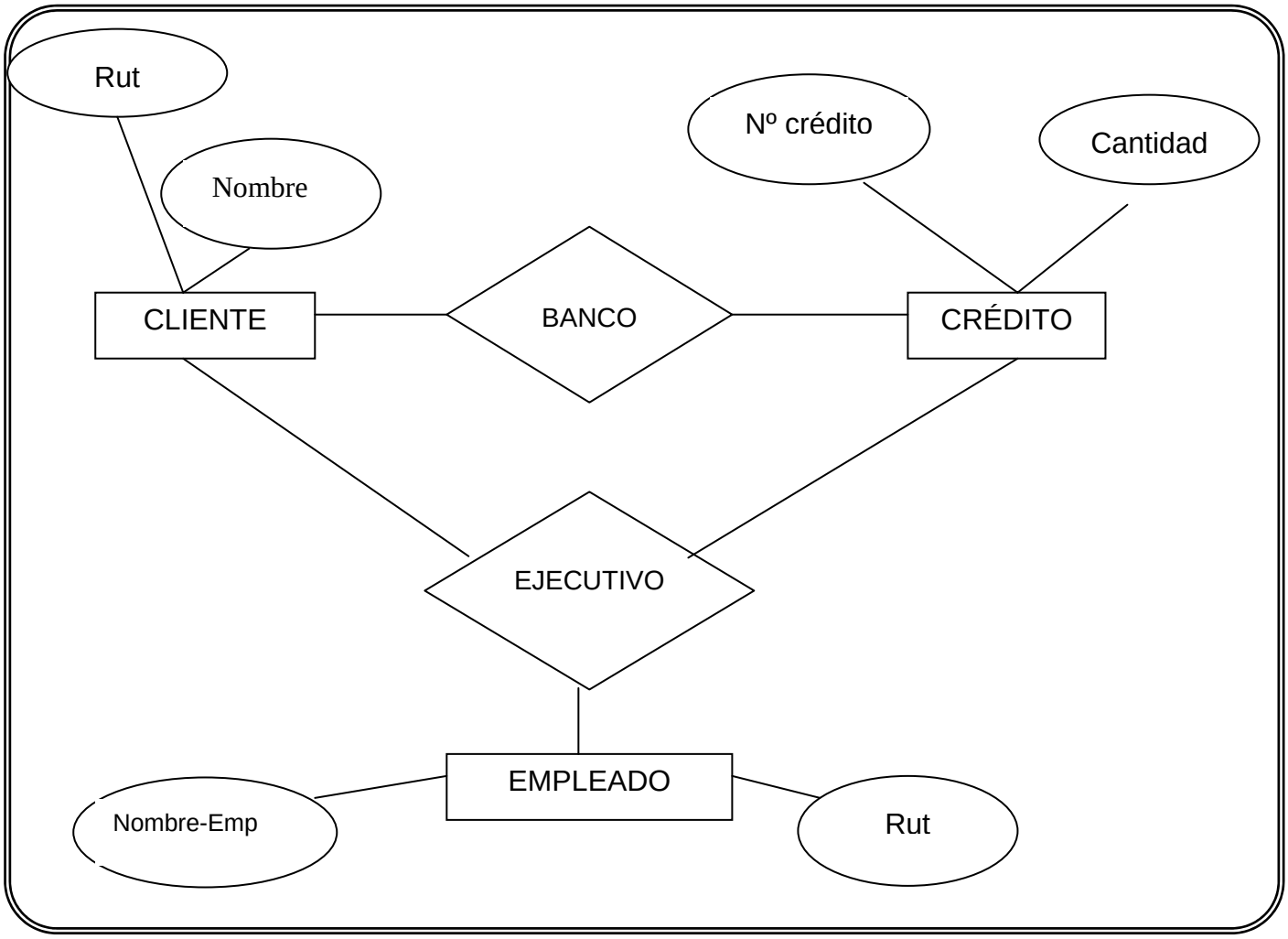
El objetivo primordial en la agregación será el establecer relaciones entre conjuntos de entidades agrupadas.

En el ejemplo que mostraremos se presenta una base de datos que posee información acerca de Clientes y Créditos.

Cada cliente va a tener como responsable a un ejecutivo del banco que será el encargado del crédito para el cliente en particular

Usando el modelo Entidad-Relación se obtiene un diagrama E-R como el siguiente:

Ejemplo Nº 23:

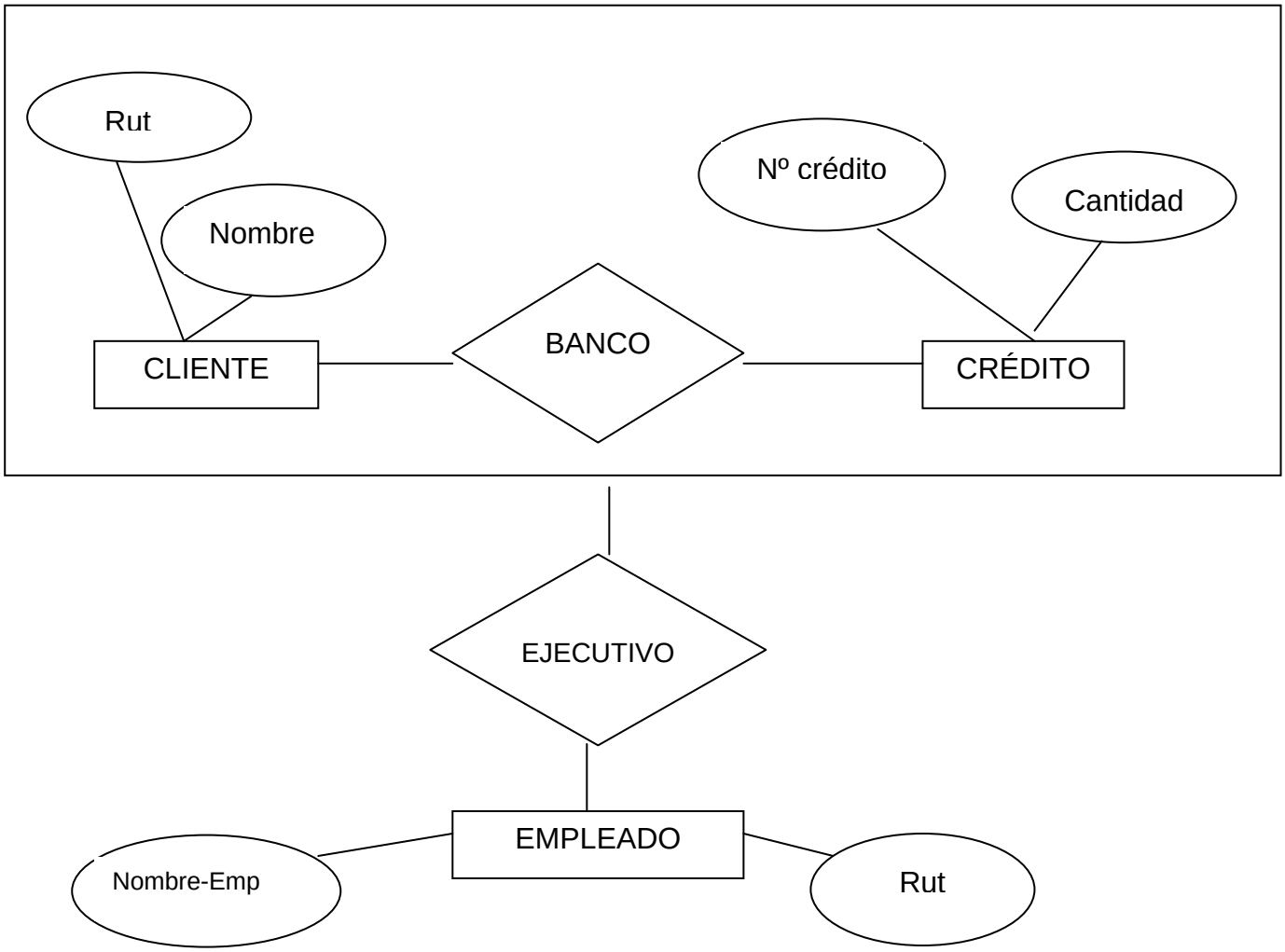



En el ejemplo se puede observar que las entidades BANCO y EJECUTIVO, se pueden combinar en un conjunto de relaciones simples. Ahora si se combinan los conjuntos de relaciones BANCO y EJECUTIVO, especifica que un ejecutivo debe estar asignado a Cliente y Crédito, lo que no puede ser debido a que habría duplicación de datos de acceso a la misma información por ambos camino.

En esta figura resultante se produce redundancia de datos, ya que tanto la entidad Cliente como Crédito están en las relaciones EJECUTIVO y BANCO

Para resolver este problema se usa la Agregación, a través del cual las relaciones se tratan como entidades de nivel más alto. Así para el ejemplo anterior se considera el conjunto de relaciones BANCO y los conjuntos de relaciones Cliente y Crédito como un conjunto de entidades llamado BANCO. El ejemplo corregido quedaría de la siguiente forma:

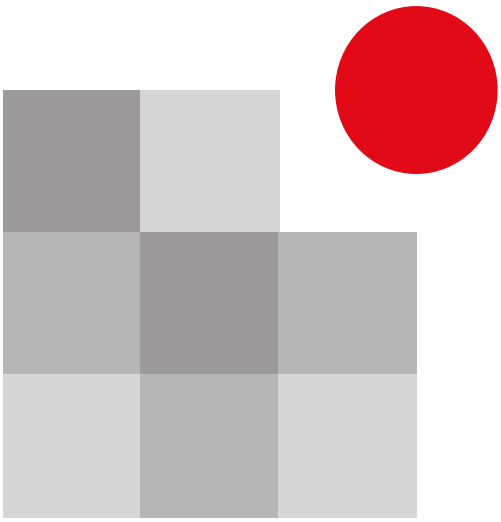
Ejemplo Nº 19




En el ejemplo se muestra la figura correcta mediante la Agregación. Se considera el conjunto de relaciones BANCO y los conjuntos de relaciones Cliente y Crédito como un conjunto de entidades llamado BANCO. Tal conjunto de entidades se trata de la misma manera que cualquier otro conjunto de entidades.

Realice ejercicios nº 25 al nº 30

SISTEMAS DE INFORMACIÓN II



IPLACEX
instituto profesional

UNIDAD III

# ESQUEMA MODELO RELACIONAL

# CLASE 01

## 1. CONCEPTOS DEL MODELO RELACIONAL

El modelo relacional tiene su base en las matemáticas, particularmente del álgebra y del cálculo.

Este modelo representa la base de datos como una colección de relaciones, es decir, cada relación se asimila a una tabla de valores, o hasta cierto punto a un fichero de registros.

En el modelo relacional, cada fila de la tabla representa un hecho que, normalmente, se corresponde con una entidad. El nombre de la tabla y los nombres de las columnas ayudan a interpretar el significado de los valores que están en cada fila.

El objetivo fundamental del modelo relacional es "mantener la independencia de la estructura lógica respecto al modo de almacenamiento y a otras características de tipo físico".

Hay también una serie de objetivos propuestos por Cood[1] (1990) que se pueden resumir en los siguientes:

### a) Independencia física.

El modo en que se almacenan los datos no influye en su manipulación lógica, y por tanto, no es necesario modificar los programas por cambios en el almacenamiento físico. (Codd concede mucha importancia a este aspecto de Independencia de ordenación, independencia de indexación e independencia en criterios de acceso).

### b) Independencia Lógica.

La inserción, modificación o eliminación de objetos en la base de datos no debe repercutir en los programas y/o usuarios que estén accediendo al subconjunto parcial de la base de datos (vistas).

### c) Flexibilidad.

Poder presentar a cada usuario los datos de la forma que prefiera.

[1] E.F.Cood propuso el modelo relacional a fines de los años sesenta, este trabajo llevó a que ganara un prestigioso premio. Con su propuesta surgieron varios proyectos para desarrollar Sistemas de datos relacionales.

d) Uniformidad.

Las estructuras lógicas de datos presentan un estado uniforme, lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios.

e) Sencillez.

Las características anteriores, así como unos lenguajes de usuario muy sencillos, producen como resultado que el modelo de datos relacional sea fácil de comprender y de utilizar por parte del usuario final.

Ejemplo

Una tabla de nombre Alumno, este nombre es porque los datos que hay en las filas representan hechos que son características de una entidad Alumno, los nombres de la columna son: Nombre, Número-Alumno, Dirección) estos datos especifican los valores que se deben interpretar como datos en las filas, teniendo en cuenta la columna en la que se encuentra cada valor.

Tabla ALUMNO → Nombre de Relación

Atributos

| NOMBRE | NUMERO-ALUMNO | DIRECCION |
|---|---|---|
| BARBARA MEZA | 20051 | 1 NORTE #256 |
| CAROLINA FLORES | 20052 | AVDA. SAN MIGUEL #155 |
| JUAN PEREZ | 20053 | 18 PONIENTE # 458 |

Tuplas

En la terminología formal del modelo relacional, una fila se denomina tupla, una cabecera de columnas es un atributo y la tabla se denomina relación. El tipo de datos que describe los tipos de valores que pueden aparecer en cada columna se denomina Dominio.

Para tener una mayor claridad de los conceptos y tener una mejor precisión sobre ellos es que a continuación se definirán:

Un dominio es un conjunto de valores indivisible[2] en lo que concierne al modelo relacional, es decir, son específicos, de un solo tipo al cual pertenecen los valores que constituyen la tabla. También es útil especificar un nombre para el dominio que ayude a interpretar sus valores.

Por ejemplo

- Nombres: conjuntos de nombres de personas
- Promedio de notas: valores posibles de los promedios de notas calculados.
- Edades de empleados: edades posibles de los empleados de una empresa

El ejemplo indica una definición lógica de dominio, también debe especificarse un tipo de formato para cada dominio, por ejemplo en el caso de las edades de los empleados, deben ser números que estén entre los 15 y 80 años.

Cabe destacar que distintos atributos pueden aplicarse sobre un mismo dominio. A modo de ejemplo, si en una relación existe un atributo fecha-nacimiento, y otro atributo fecha-matrimonio, ambos pueden aplicarse sobre un mismo dominio llamado fechas.

Un esquema de relación se compone de un nombre de relación R y una lista de atributos, cada atributo es el nombre de un papel desempeñado por algún dominio en el esquema de la relación. Un esquema de relación sirve para describir una relación. En otras palabras, en la tabla a cada columna se le llama atributo y al conjunto de atributos se le denota: esquema de la relación.

Por ejemplo un esquema de relación sería:

Alumno (Nombre, Numero-alumno, Dirección), en donde, Alumno es el nombre de la relación, la cual tiene tres atributos.

Es importante mencionar que una relación se dice que está definida por Extensión, si lo es por las tuplas existentes en un instante dado. Esta definición es de carácter dinámica por cuanto varía en el tiempo, pudiendo eliminarse o modificarse las tuplas existentes, e incorporarse nuevas tuplas. Es decir, relación también llamado por extensión es una tabla de filas (tuplas) y columnas (atributos), que puede ser modificada, dependiendo de la necesidad de la empresa, se puede actualizar la información, eliminar, agregar, etc..

[2] Indivisible, que no se pueden separar, siguen una misma línea de valores en los datos.

En el modelo relacional, el concepto de clave está definido de una manera similar al concepto identificador del modelo ER, una clave de una relación es un conjunto de atributos de la relación, que identifica de manera única a cada tupla de cada extensión de esa relación. Esto significa que dos o más tuplas no pueden tener el mismo valor como atributo identificador.

Ahora bien una relación puede tener más de una clave y cada clave se denomina clave candidata.

Ejemplo

Una tabla Alumno puede tener dos claves candidatas, por ejemplo, la primera el Numero de Matricula, y la segunda una clave compuesta por Apellido y Nombre. Es normal designar una de las claves como clave primaria de la relación. Normalmente se subrayan los atributos que contenga la clave primaria.

| NUMERO-MATRICULA | NOMBRE | APELLIDO |
|---|---|---|

Clave Primaria

También se puede decir que aquellos atributos de una relación R que no siendo claves de dicha relación lo sean de alguna otra relación, se denominan claves foráneas. Esto es, un atributo de una relación R se considera como clave foránea si dicho atributo es clave de otra relación.

Ejemplo

La relación CARRERA (numero-alumno, Rut, nombre, apellido, dirección,) podía también haberse especificado con la siguiente estructura:

CARRERA (Nº-carrera, numero-alumno, Rut, nombre, apellido, dirección)

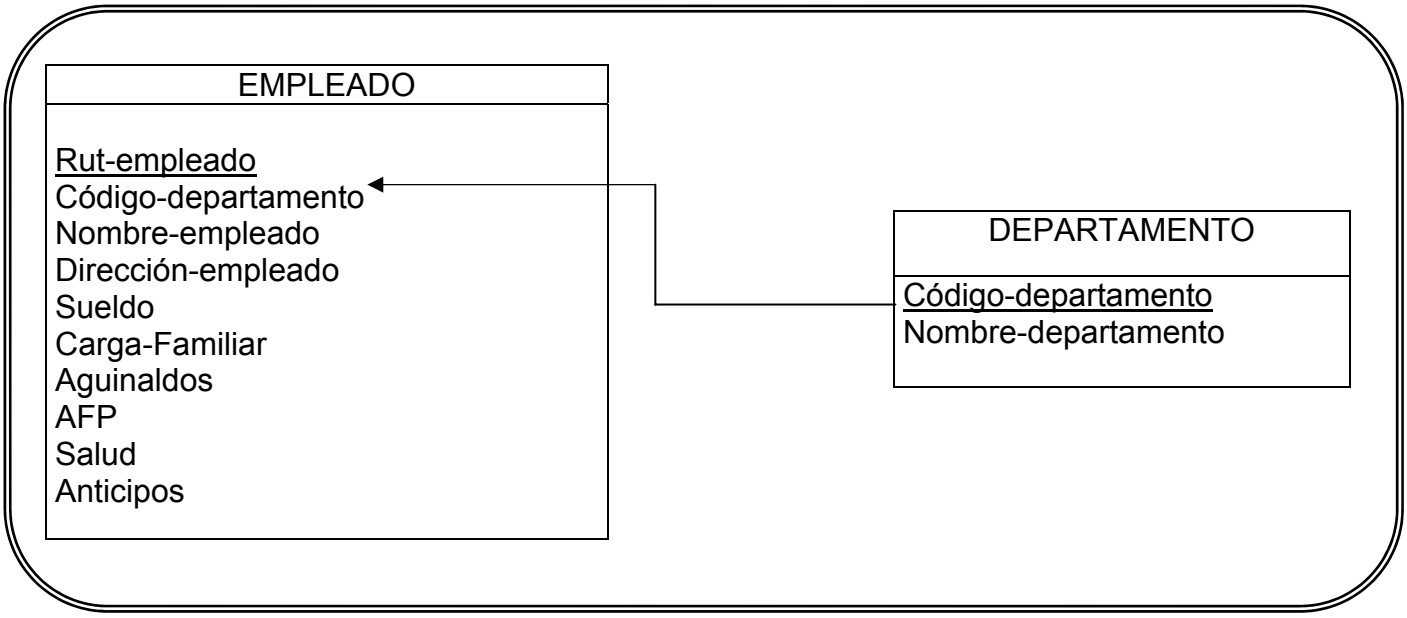
Tanto en una como la otra estructura que se muestran, los atributos numero-alumno y Rut son claves foráneas, con la diferencia que en un caso son parte de la clave de la relación CARRERA, en tanto que en el otro caso no lo son, la clave es Nº-carrera.

# CLASE 02

## 2. DIAGRAMA Y ESQUEMA RELACIONAL

Para representar gráficamente una realidad dada bajo el modelo relacional existen distintas notaciones, concordando la mayoría de ellos en representar las notaciones mediante rectángulos. Pero el diagrama que se usará se conoce también con el nombre de Diagrama de Bachman[3].

Ejemplo




El ejemplo muestra un Diagrama de Bachman en donde, Código-departamento de la tabla DEPARTAMENTO es igual a Código-departamento de la tabla EMPLEADO, lo que implica que es la clave principal de la tabla DEPARTAMENTO y clave secundaria para la tabla EMPLEADO. Esta clave permite que a través de ella se pueda acceder a la tabla DEPARTAMENTO, y poder recoger la información que se requiere.

### 2.1. Transformación del Modelo Entidad-Relación al Modelo Relacional.

¿Por qué es necesaria la transformación a Modelo Relacional?, porque las Base de Datos comerciales están basadas en el modelo relacional, además son más sencillas de operar atributos monovaluados[4] y posee lenguajes poderosos de consulta como lo es SQL.

[3] Bachman, (1990), crea la estructura de datos del diagrama que lleva su nombre. En este modelo se describen los tipos de entidades, tipos de interrelaciones representados en forma de nodos

[4] Atributos monovaluados: son aquellos en donde el atributo tiene un solo valor, como por ejemplo el rut de una persona.

SQL es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos informática. El nombre de "SQL" es una abreviación de Structured Query Language (Lenguaje Estructurado de Consultas). Como su nombre indica, SQL es un lenguaje informático que se utiliza para interactuar con un tipo de bases de datos, llamado base de datos relacional. SQL es un lenguaje de bases de datos relacionales, y ha llegado a ser popular junto con el modelo relacional de base de datos.

El punto de partida es un esquema entidad relación y el resultado de la correspondencia o transformación es el esquema relacional. Esta correspondencia es un conjunto de relaciones en donde cada relación tiene una clave primaria.

Existe una forma habitual de representar la realidad usando el modelo relacional, como se puede observar cuando se traducen diagramas Entidad-Relación a esquemas relacionales. Los diagramas E-R vienen definidos fundamentalmente por entidades y relaciones.

Cada entidad se transforma en una tabla con los mismos atributos, se agregan claves donde sea necesario y cada relación se transforma en una tabla en que los atributos son las claves de cada entidad participante.

A continuación, se indica como se representa en el modelo relacional cada uno de estos conceptos:

a) Entidad: como se ha visto anteriormente, una entidad se representa por una relación definida por el conjunto de la entidad y los dominios asociados a estos atributos. Cada tupla de esta relación corresponderá a una ocurrencia de la entidad.

b) Relación: una relación R en la que forman parte las entidades E1, E2, ..., En, se representa por una relación definida por el conjunto de los atributos identificadores de las n entidades y los dominios correspondientes a dichos atributos (renombrando atributos si fuera necesario).

- Transformación de Atributos de Entidades.

Cada atributo de una entidad se transforma en un atributo de la relación (columna de la tabla), según se señala en las siguientes tres subreglas:

a) Atributos identificadores principales (AIP): pasan a ser la clave primaria de la relación y en SQL se utiliza la cláusula PRIMARY KEY. Esta cláusula de SQL se utiliza para consultar en la base de datos específicamente por la clave primaria y a través de ella acceder a toda la información de la tabla.

b) Atributos identificadores alternativos: se transforman en claves alternativas, son claves que si no se accede por la clave primaria existen estas otras que permiten acceder ella siendo también únicas. en SQL se utiliza la cláusula UNIQUE.

c) Atributos no identificadores: serán columnas normales que podrán tomar valor nulo salvo que se indique lo contrario.

Realice ejercicios nº 1 al nº 4

Ejemplo

Tabla

NSS
NOMBRE
EMPLEADO APELLIDO
SALARIO

Relación que se forma:

EMPLEADO( NSS, NOMBRE, APELLIDO, SALARIO)

Siendo NSS: un número de servicio social que algunas empresas asignan a cada empleado por lo tanto es una clave única de identificación.

La transformación a Modelo Relacional es un paso bastante simple, se transforma cada entidad en una relación. Los atributos y la clave primaria de la entidad se convierten en los atributos y la clave de la relación.

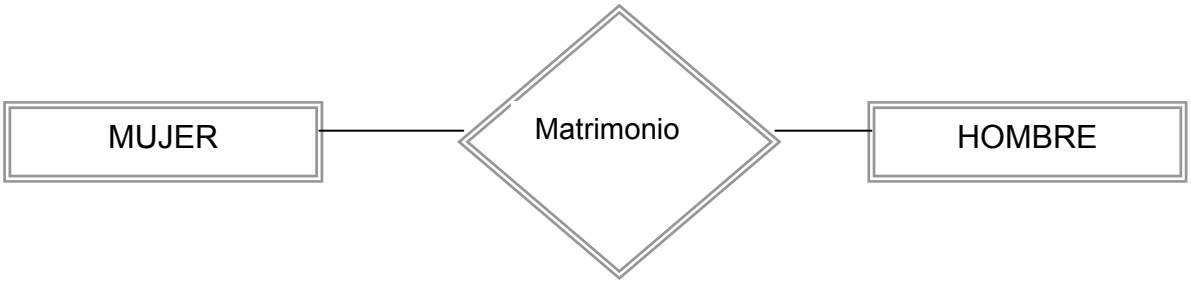
Para poder transformar una entidad de MER[5] a MR[6] y haya correspondencia entre ambos, se deben considerar los siguientes elementos:

- Una entidad en el modelo entidad relación es equivalente a un modelo relacional.
- Cada tupla de la relación representa una ocurrencia de una entidad. La Clave de una relación no es sino la clave de la entidad.
- Si el conjunto de ocurrencias es identificable por su asociación con otra entidad, entonces la relación deberá incluir entre sus atributos aquellos que son claves de las entidades.

[5] MER: Modelo Entidad Relación
[6] MR: Modelo Relacional

Ejemplo

El modelo entidad-relación sería:




La relación modelo relacional sería:

MATRIMONIO (Codigo_Hombre, Codigo_ Mujer)
HOMBRE (Codigo_Hombre, nombre, dirección, fono )
MUJER (Codigo_Mujer, nombre, fono, dirección )

En el ejemplo se puede observar que la entidad se transforma en un modelo relacional desglosado para poder entender mejor la relación y poder acceder de mejor manera a la información sin afectar a las tablas si se quiere eliminar, agregar o modificar. Si se dejara tal cual la entidad, la relación quedaría:

MATRIMONIO (Codigo_Hombre, Codigo_ Mujer, nombre-hombre, dirección, fono, nombre-mujer)

## 2.2 Transformación de interrelaciones uno a uno.

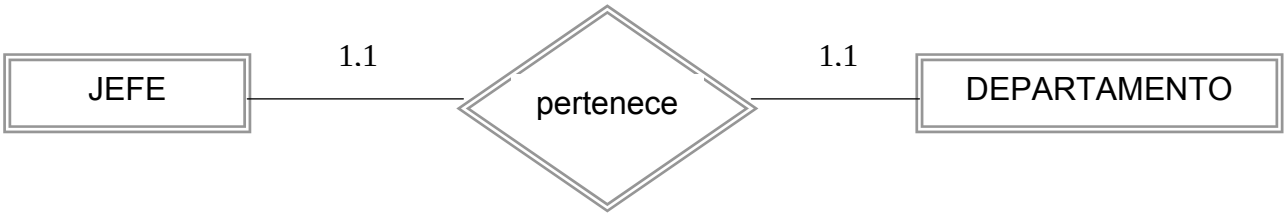
Se verán las transformaciones de manera individual, este proceso también es influido por las cardinalidades mínimas[7] de las dos entidades que participan en la interrelación.

Existen dos posibilidades para la integración en una relación, las cuales son:

1.- Las dos entidades tienen las mismas claves primarias. Por ejemplo se tiene que tanto JEFE como DEPARTAMENTO tienen la misma clave primaria Rut_jefe. En este caso las dos relaciones se integran en una relación combinando todos los atributos e insertando la clave primaria sólo una vez.

[7] Cardinalidad Mínima: es el número mínimo de veces con que una ocurrencia de una entidad participa en una relación determinada.

Ejemplo




Las relaciones son:

JEFE (Rut_jefe, Nombre_jefe, Cargo)
DEPARTAMENTO (Rut_jefe, Nombre_departamento)

La integración quedaría:

PERTENECE-DEPARTAMENTO (Rut-jefe, Nombre-jefe, Cargo, Nombre-departamento)

## CLASE 03

2.- En este segundo caso es cuando las entidades tienen dos claves primarias distintas, por ejemplo JEFE y DEPARTAMENTO tienen diferentes claves primarias, como Rut_jefe, y Codigo_departamento respectivamente. También aquí se integran las entidades combinando todos los atributos incluyendo las claves primarias de ambos.

Una de las dos claves será la clave primaria de la relación resultante.

La relación resultante será:

PERTENECE_DEPARTAMENTO (Rut_jefe, Nombre_jefe, Cargo, Codigo_departamento Nombre_departamento)

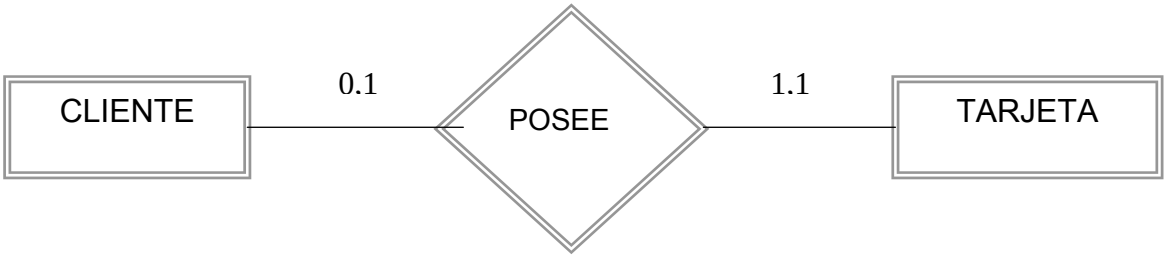
La clave primaria elegida es Rut-jefe, ya que es la clave que unifica a una persona . es decir dos personas nunca tendrán un mismo rut.

También existe la opción de realizar una integración aparte, la cual se usa cuando una o las dos entidades tienen una participación parcial, es decir, cuando una entidad participa no del todo en una relación, sólo si es necesario.

Un ejemplo sobre este tipo de integración es cuando los clientes de un banco se les presenta cero o una tarjeta de crédito. Cada tarjeta de crédito debe pertenecer a un cliente, pero puede ser que un cliente no quiera tener tarjeta de crédito. En este caso las relaciones CLIENTE y TARJETA_CREDITO ya han sido creadas

Se define una relación adicional llamada POSEE_TARJETA (numero_cliente, tipo_tarjeta, numero_tarjeta), usando la clave primaria de las dos relaciones. Cualquiera de los atributos de POSEE_TARJETA puede ser clave primaria. Se puede tomar la primera posibilidad mencionada antes y representar todo en una sola relación, resultando en este caso tomar numero_cliente como clave primaria de la relación integrada. Aquellos clientes que no posean la tarjeta de crédito tendrán valores nulos de los atributos tipo_tarjeta y numero_tarjeta, es por esto que no se puede seleccionar a estos atributos como claves primarias de la relación integrada, ya que en ese caso los clientes sin tarjetas no podrían ser representados.

Ejemplo




Las relaciones son:

CLIENTE (numero-cliente, nombre-cliente)
TARJETA-CREDITO (tipo-tarjeta, numero-tarjeta, crédito)
POSEE-TARJETA (tipo-tarjeta, numero-tarjeta, numero-cliente)

Y relación integrada sería:

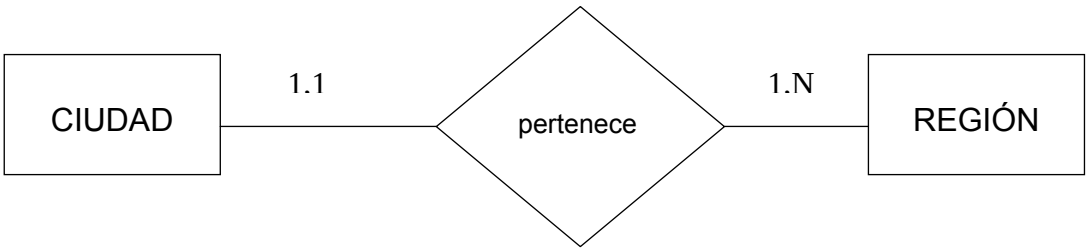
POSEE-TARJETA (numero-cliente, nombre-cliente, tipo-tarjeta, numero-tarjeta, crédito)

## 2.3 Transformación de Interrelaciones Uno a Muchos.

Una interrelación es de uno a muchos entre las tablas A y B cuando una clave de la tabla A posee varios elementos relacionados en la tabla B y cuando una clave de la tabla B posee un único elemento relacionado en la tabla A.

Para este proceso de integración se tienen dos casos, primero la entidad en el lado de muchos tiene una participación obligatoria, hay una interrelación de uno a muchos entre REGIÓN y CIUDAD. Ahora ciudad tiene una participación total, por lo que la clave primaria de REGIÓN se incluye en la relación CIUDAD

### Ejemplo



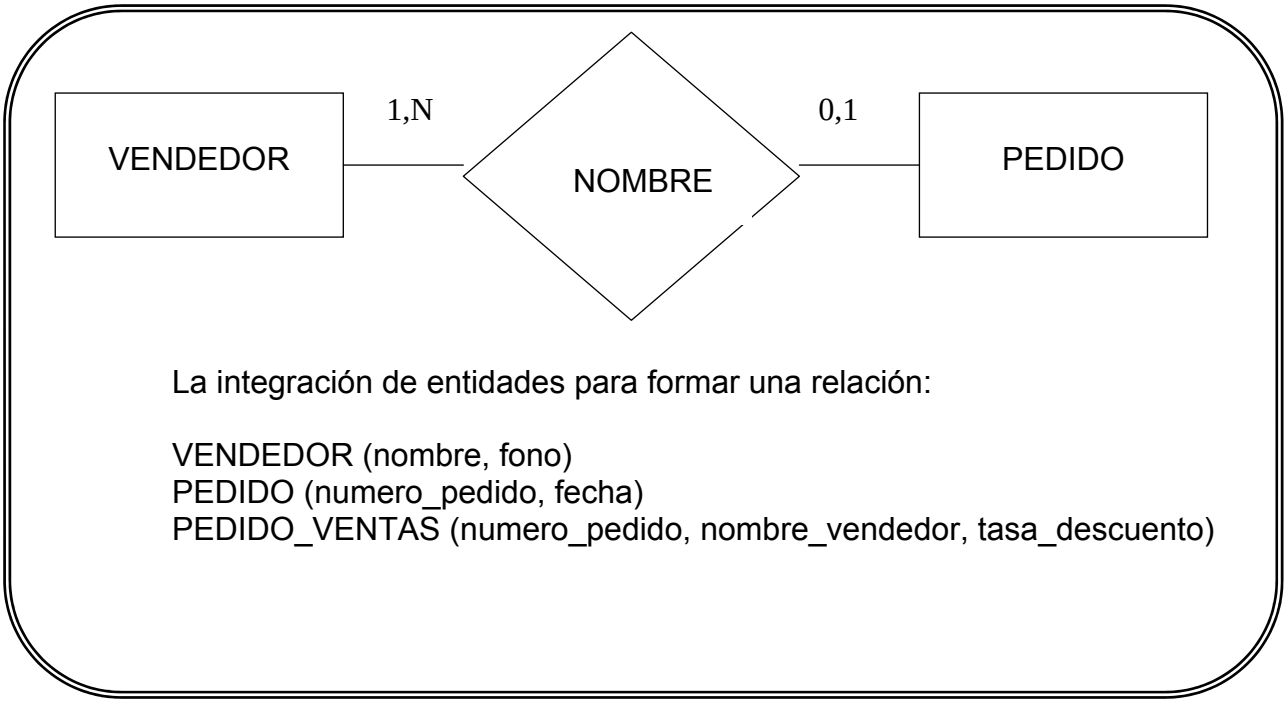

Para este Caso 1, la relación integrada sería:

CIUDAD (nombre_ciudad, nombre_estado, habitantes)
REGIÓN (nombre_región, intendente, habitantes)

A continuación, se realiza el análisis de la relación entre la tabla de VENDEDOR y la tabla de PEDIDOS. Un vendedor puede realizar varios pedidos pero un pedido pertenece a un único vendedor, esa es la traducción de la relación uno a varios y se representa 1: n.

En este último caso la entidad en el lado de muchos tiene participación parcial. Existe una relación entre vendedor y pedido. Ahora los pedidos pueden hacerse a través de los vendedores aplicando un descuento, pero además a los pedidos sin vendedores no se aplica tal tasa de descuento. Con esta condición existe la posibilidad de valores nulos para los atributos nombre_vendedor y tasa_descuento.

Ejemplo




En este caso, como no se admiten valores nulos, se adopta la mejor opción que es establecer tres relaciones (vendedor, pedido, pedido_ventas). Nótese que las dos relaciones PEDIDO y PEDIDO_VENTAS describen pedidos, la primera relación incumbe a todos los pedidos, en cambio la relación PEDIDO_VENTAS contiene en forma de un subconjunto sólo a aquellos pedidos hechos por vendedores.

## 2.4. Transformación de Interrelaciones Muchos es a Muchos.

Se transforma en una nueva relación que tendrá como clave principal la concatenación de las claves de los tipos de entidades participantes.

Cada atributo de la clave primaria será clave externa respecto de una de las relaciones que representan las entidades participantes.

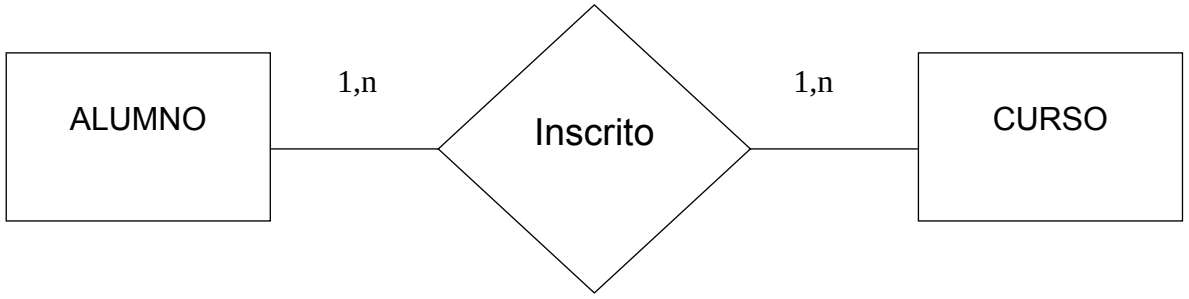
Es imposible diferenciar en el esquema relacional las relaciones que provienen de tipos de entidad y cuales son las que provienen de tipos de interrelaciones.

A cada uno de los atributos que forman la clave primaria de una relación derivada de un tipo de interrelación N:M son clave foránea respecto de cada una de las relaciones derivadas de los tipos de entidad que relaciona.

Como un ejemplo se tiene una relación INSCRITO de muchos es a muchos entre dos entidades ALUMNO y CURSO. Se crea una relación nueva que tendrá como clave

primaria la combinación de atributos que constituyen las claves primarias de ambas entidades.

Ejemplo




La relación integrada sería:

ALUMNO (codigo_alumno, nombre, promedio)
CURSO (codigo_curso, nombre_curso)
INSCRITO (codigo_alumno, codigo_curso, semestre, nota)

Se observa que se modela como una nueva relación INSCRITO con ALUMNO y CURSO, en donde se tiene como clave primaria a codigo_alumno y codigo_curso con semestre y nota como atributos. Con codigo_alumno como clave foránea que referencia a ALUMNO y codigo_curso como clave foránea que referencia a CURSO.

En la práctica quedará lo siguiente:

INSCRITO (codigo_alumno, codigo_curso, semestre, nota)

Que permite demostrar que a través de las claves foráneas de la relación inscrito que a su vez son las claves primarias de las relaciones CURSO y ALUMNO se puede acceder a estas relaciones respectivas para trabajar con la información

Realice ejercicios nº 5 al nº 9

# CLASE 04

## 3. REGLAS DE INTEGRIDAD

Para poder garantizar que los datos almacenados en una estructura son correctos, se deben estudiar las reglas de integridad.

Existen dos reglas de integridad muy importantes que son restricciones que se deben cumplir en todas las bases de datos relacionales y en todos sus estados o instancias.

Se consideran tres tipos de restricciones como parte del modelo relacional:

a) Restricciones de Clave
b) Integridad de Entidades
c) Integridad Referencial

A continuación, se analizan cada una de ellas.

a) Restricciones de Clave: son aquellas que especifican las claves candidatas de cada relación. Los valores de las claves candidatas deben ser únicos para cada tupla en cualquier caso de esa relación.

Ejemplo

CLAVE

ALUMNO

| RUT | NOMBRE | EDAD |
|---|---|---|
| 11951455-K | Carla Solari | 30 |
| 13305019-1 | Edmundo Meza | 30 |

En el ejemplo Rut es la clave para la tabla alumno y ésta constituye una restricción en el esquema de base de datos puesto que solo a través de Rut se podrá acceder a la información de la tabla.

b) Restricción de Integridad de Entidades: esta restricción establece que ningún valor de clave primaria puede ser nulo. Esto se debe a que el valor de la clave primaria se utiliza para identificar las tuplas individuales de una relación. Si hubiera un valor nulo para la clave primaria implicaría que no podría identificarse alguna tupla.

Las claves primarias compuestas deben de ser no nulas en su totalidad, es decir, ninguno de sus atributos puede ser nulo.

Ejemplo

CLAVE

ALUMNO

| RUT | NOMBRE | EDAD |
|---|---|---|
| | Carla Solari | 30 |
| 13305019-1 | Edmundo Meza | 30 |

En el ejemplo se puede analizar que si la clave primaria tiene por valor nulo[8], entonces no se podría distinguir cuál es el alumno a quien corresponden los datos.

Es importante destacar la razón por la cual ningún valor de la clave primaria puede ser NULO. Esto es porque el valor de la clave primaria se usa para identificar las tuplas individuales de una relación; permitir valores nulos para la clave primaria implica que no se pueda identificar algunas tuplas.

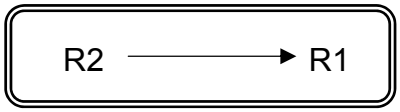
c) Restricción de Integridad Referencial: este tipo de restricción se utiliza para mantener la coherencia entre las tuplas de las dos relaciones, en otras palabras, establece que una tupla de una relación que haga referencia a otra relación debe referirse a una tupla existente en esa relación.

Por ejemplo, en una base de datos académica, existe un alumno inscrito a una clase inexistente, o un curso impartido por un profesor inexistente.

Para poder definir de mejor manera la integridad referencial, es importante conocer el concepto de Clave Ajena.

Una Clave Ajena es un conjunto de atributos de una relación R2 cuyos valores o son completamente nulos o coinciden con los de la clave primaria de una relación R1.

De un modo gráfico se representa mediante una flecha entre las dos relaciones que se etiqueta por el conjunto de atributos que la definen. Esto es:




[8] Valor Nulo: significa valor desconocido o inexistente.

Ejemplo

Se tiene la tabla CURSO con las siguientes claves primarias (Numero_curso, Nombre_Alumno), el atributo Nombre_Alumno se refiere al alumno que pertenece al Curso, por lo tanto, Nombre_Alumno sería la clave ajena de CURSO, refiriéndose a la relación ALUMNO. Esto significa que un valor de Nombre_Alumno en cualquier tupla de la relación CURSO debe coincidir con un valor de la clave primaria de ALUMNO, el atributo Nombre, de alguna tupla de la relación ALUMNO, o bien, ser un valor nulo. Pero se debe recordar que si existiese algún valor nulo en Nombre_Alumno se violaría la regla de la integridad de entidades de la relación CURSO.

Ahora bien, si se tuviera una clave ajena denominada Num_sala en CURSO, especificando el número de la sala en que se realizarán las clases del Curso, y que se refiera a la clave primaria de una relación llamada SALA, habría otra vez una restricción referencial, con la salvedad que los valores nulos de Num_sala de SALA (especificando que el número de la sala es desconocido)

Realice ejercicios nº 10 al nº 11

# CLASE 05

## 4. VALORES NULOS EN EL MODELO RELACIONAL

Si bien los valores nulos no son un concepto exclusivo del modelo relacional, se puede definir el valor nulo o valor ausente como una marca utilizada para representar información desconocida, inaplicable, etc. Se dice que el valor nulo representa la falta de información

Un valor nulo es un indicador que dice al usuario, que el dato falta o no es aplicable.

Ejemplo

| Rut | Nombre | Dirección |
|---|---|---|
| 35784843 | María Gutiérrez | ¿? |

La siguiente tupla:

{(Rut, 35784843), (nombre, María Gutiérrez), (dirección, ?)}

En esta tupla el atributo dirección tiene valor nulo indicando que se desconoce la dirección de esta persona.

Dentro del modelo relacional existen algunas diferencias cuando los atributos tienen valores nulos y a continuación se mencionan:

a) Dominio.

En el concepto de dominio, cualquier símbolo que se utilice para los atributos que puedan tomar valor nulo lo cambia significativamente, y se reconoce como un elemento más, dentro de la relación.

b) Relación.

No puede haber tuplas duplicadas, ahora una tupla es duplicada de la otra cuando todos los atributos que son nulos en una lo son también en la otra, y todos los que no son nulos tienen en la otra el mismo valor.

c) Superclave.

Un atributo es una superclave cuando sus valores no nulos son todos distintos. Si el atributo es compuesto, se entiende que no es nulo cuando todos sus componentes lo sean.

d) Falta de Información

Se requiere de una forma de controlar cuando se tiene una base de datos en la que falta información porque no se cuenta con ella.

Ahora según Codd la idea fundamental es que cualquier situación en que la fila y columna en la base de datos se podría marcar como nula con la siguiente anotación "nula" (siempre y cuando no lo impidan restricciones de integridad), que qeuiere decir que en esa posición falta información.

## 4.1. Otros tipos de Valores Nulos

Hasta ahora se ha limitado a examinar un solo tipo de valor nulo, a saber, "valor desconocido". Sin embargo, pueden existir varios otros tipos; por ejemplo, "la propiedad no es aplicable", "el valor no existe", "el valor no está definido".

En cuanto a las operaciones aritméticas con valores nulos, se toma como nulo el resultado de suma, multiplicar, restar o dividir un valor nulo con cualquier otro valor.

Otro aspecto que hay que tener en cuenta es la realización de operaciones de agregación con el fin de aplicar funciones estadísticas, como frecuencias, varianza, media, etc. Para estos casos hay que ver claramente si se han de tener o no en cuenta los valores nulos al efectuar los cálculos, ya que ocurrirán incoherencia con los resultados y no serán entregados datos correctos.

El gran problema de los valores nulos es que ningún producto relacional hasta la fecha los ha establecido de manera coherente y satisfactoria; no hay reglas que funcionen para todos los casos iguales, por ejemplo, hay veces que no se tienen en cuenta los valores nulos, como para las operaciones de agrupamiento, pero en otros casos sí, como en la ordenación o proyección.

## 4.2. Valores por Defecto en lugar de Valores Nulos

Hay algunos autores como Date (1990), que propugnan evitar los valores nulos, mediante la utilización de valores por defecto. Pero no es un buen enfoque ya que no resuelve el problema de valor nulo si no que facilita un medio para representar información desconocida.

Además las distintas técnicas para utilizar los valores nulos por defecto quedan concentradas en los programas, con pocas posibilidades de ser uniformes, sistemáticas y estar bien documentadas.

Cada valor desconocido se trata, en definitiva, como si fuera cualquier otro valor.

Mientras no haya solución al problema del valor nulo será mejor que el administrador de base de datos busque la mejor solución para tratar como se manejará este valor.

## 5. PROCESO DE NORMALIZACIÓN

El diseño de esquemas para generar bases de datos relacionales debe considerar el objetivo de almacenar información sin redundancia innecesaria, pero que a la vez permitan recuperar información fácilmente. Una técnica consiste en diseñar esquemas que tengan una forma normal adecuada.

Por ejemplo

| INSCRIPCIÓN | | | | |
|---|---|---|---|---|
| 93415091 | Raúl | San Javier | Administración | 5 |
| 93415091 | Raúl | San Javier | Economía | 4 |
| 95413014 | Pedro | Talca | Administración | 5 |
| 93415091 | Raúl | Villa Alegre | Cálculo | 4 |

La tabla de inscripción presentada tiene la siguiente estructura: INSCRIPCIÓN (número_matrícula, nombre, ciudad, código_asignatura, número_crédito)

¿Cree usted que está bien estructurado? Si es así, estaba equivocado, puesto que hay dos números de matrícula (uno se repite).

La clave es número_matrícula, código_asignatura, aún así está mal estructurado ya que no hay dependencia funcional completa.

De esto se trata el proceso de normalización. En el caso planteado la tabla no está normalizada, puesto que presenta inconsistencias, tales como:

a) No existe dependencia funcional completa.
b) Inconsistencia de pérdidas de información al sacar a un alumno. Por ejemplo: Pedro, o bien agregar otro alumno.

Resumiendo, la normalización es eliminar o reducir estas inconsistencias que se presentan cuando se actualiza una base de datos.

Al normalizar esta tabla quedaría del siguiente modo:

INSCRIPCIÓN (número_matrícula, código_asignatura)
ALUMNO (número_matrícula, nombre_alumno, ciudad_alumno)
ASIGNATURA (código_asignatura, número_crédito)

Ahora sí se puede señalar que existe dependencia funcional completa.

| INSCRIPCIÓN | |
|---|---|
| 93415091 | Administración |
| 93415091 | Economía |
| 95413014 | Administración |
| 93415091 | Cálculo |

| ALUMNO | | |
|---|---|---|
| 93415091 | Raúl | San Javier |
| 95413014 | Pedro | Talca |

| ASIGNATURA | |
|---|---|
| Administración | 5 |
| Economía | 4 |
| Cálculo | 4 |

Las propiedades indeseables que trae un mal diseño son básicamente

- Repetición de información
- Incapacidad para representar cierta información
- Pérdida de información.

Las formas normales, definidas en la teoría relacional, permiten evitar que estas propiedades indeseables aparezcan en una base de datos basada en un esquema mal diseñado. Un esquema debe estar a lo menos en tercera forma normal, para que sea aceptable.

Hay que considerar que las reglas de normalización están dirigidas a la prevención de anomalías de actualización e inconsistencias en los datos. Ellas no reflejan ninguna consideración de rendimiento. En cierta forma pueden ser visualizados como orientadas por el supuesto de que todos los atributos no clave serán actualizados frecuentemente.

Las relaciones tal como han sido definidas pueden llegar a tener propiedades no deseadas, como sería la inconsistencia en los datos de las relaciones.

Por ejemplo

Que una misma persona aparezca en una misma relación o en más de una, con dos direcciones diferentes.

Este problema puede surgir cuando se actualizan relaciones, ya sea porque en una misma relación dicha persona aparece más de una vez ó porque esta persona con su dirección aparece en más de una relación y sólo se actualiza una tupla de la relación y para el segundo caso sólo se actualiza la tupla de una relación y no la tupla de la segunda relación.

Ejemplo

Sean las relaciones PROVEEDOR, PRODUCTO Y PEDIDO con las siguientes estructuras:

PROVEEDOR(numero_proveedor, nombre_proveedor, cuidad)
PRODUCTO(numero_producto, nombre_producto, color)
PEDIDO (numero_proveedor, numero_producto, cantidad)

En la relación PROVEEDOR se puede ver que los atributos nombre_proveedor, ciudad, dependen funcionalmente de numero_proveedor. Es decir, para un numero_proveedor dado, existe un único valor para cada uno de los tres atributos. Decir que el atributo ciudad tiene Dependencia Funcional (DF) de numero_proveedor, equivale a afirmar que un proveedor debe estar necesariamente localizado en una única ciudad.

Se tiene la siguiente relación:

ESCRIBE (autor, nacionalidad, cod_libro, titulo, editorial, año).

Presenta varios problemas:

1) Gran cantidad de redundancia. La nacionalidad del autor se repite en cada tupla del mismo. Cuando un libro tiene más de un autor la editorial y el año se repiten también.

2) Anomalías de modificación. Puede ocurrir que se modifique el nombre de la editorial en una fila sin modificarla en el resto que corresponden al mismo libro.

3) Anomalías de inserción. No sería posible la inserción de un autor del que no hubiera ningún libro (cod_libro, Clave primaria), tampoco podría haber obras anónimas.

4) La inserción de un libro con más de un autor obligaría a la repetición de tuplas.

5) Anomalías de borrado. Si se quiere dar de baja un libro también se perdería información de los autores y viceversa.

Ante cualquier duda de si el modelo es correcto, es preferible la aplicación de un método formal de análisis, es decir, la Teoría de la Normalización.

Es necesario formalizar una teoría en la que se estudien las mejoras que da el estar en una forma normal o en otra y donde se defina formalmente las condiciones que

debe cumplir una relación para estar en una forma o en otra. Esta teoría permitirá afrontar el diseño de una forma estricta y permitirá asegurar que el diseño estará libre de una gran parte de problemas.

Realice ejercicios nº 12 al nº 15

# CLASE 06

## 5.1. Formas Normales.

La teoría de normalización consiste en obtener esquemas relacionales que cumplan unas determinadas condiciones y se centra en las determinadas Formas Normales. Se dice que un esquema de relación está en una determinada forma normal si satisface un conjunto determinado de restricciones.

- Primera Forma Normal

Se dirá que una relación se encuentra en primera forma normal si existe una dependencia funcional de cada uno de los atributos que la conforman respecto del atributo identificador principal o clave primaria y también cuando no hay grupos repetitivos en sus atributos.

Ejemplo

ALUMNO

| RUT | NOMBRE | DIRECCIÓN | FONO |
|---|---|---|---|
| 1 | Juan | 3 oriente 256 | 256589 |
| 2 | José | 1 sur 155 | 456958 |
| 2 | José | 1 sur 155 | 451255 |
| 3 | María | Alameda 1154 | 271588 |

Sea la relación ALUMNO(rut , nombre, dirección, fono). Se sabe que todo alumno tiene una única dirección, pero puede tener asignado más de un fono. Por lo tanto se tiene que mencionar que sólo puede estar asignado un fono a un alumno.

Ahora la relación ALUMNO no está en 1FN porque no existe dependencia funcional del atributo fono con respecto a la clave Rut, ya que existe un mismo alumno con distinto fono.

También Rut ya no sería clave primaria de la relación porque no identifica las tuplas al haber más de una tupla con la misma clave. En consecuencia, la clave debiera ser la unión de los atributos Rut y Fono.

- Segunda Forma Normal

Se dirá que una relación se encuentra en 2FN cuando está en su 1FN y cada atributo que no forma parte de la clave tiene Dependencia Funcional Completa[9] (DFC) respecto de la clave.

Una relación está en 2FN si además de estar en 1FN todos los atributos que no forman parte de ninguna clave candidata suministran información acerca de la clave primaria.

Toda relación cuya clave esta formada por un solo atributo está en 2FN.

Ejemplo

Sea la siguiente relación :

CALIFICACIÓN(nº_matricula, codigo_curso, año, nota_final, nombre_alumno, dirección, nombre_curso)

Esta relación se encuentra en 1FN, lo que indica que pueden haber las siguientes anomalías:

a) Si se quisiera eliminar un codigo_curso de un alumno se perdería toda la información que acompaña al alumno.

b) Si se quiere agregar un alumno, se puede caer en el error de ingresar un mismo alumno dos veces.

c) Además de tener dependencia funcional completa ya que todos los atributos que no forman parte de la clave dependen de ella.

Para solucionar este problema se debe reformular la relación quedando de la siguiente manera:

[9] DFC: es cuando Y depende funcionalmente de X, pero no de ningún subconjunto obtenido de los posibles atributos que forman A, siendo A un Atributo que dependa de X.

CALIFICACIÓN(nº_matricula, codigo_curso, año, nota final)
CURSO(codigo_curso, nombre_curso, nombre_alumno, dirección)

Con esta nueva reestructuración se puede incorporar un nuevo alumno aún cuando no forme parte de ningún curso, se puede eliminar una calificación sin que se pierda información de ningún alumno. La relación queda en segunda forma normal.

### Ejemplo

PRESTAMO (num_socio, nombre_socio, cod_libro, fecha_prestamo, editorial, país)

Claves candidatas:

(num_socio, cod_libro)
(nombre_socio, cod_libro)

Para pasarlo a 2FN:

PRESTAMOS (num_socio, nombre_socio, cod_libro, fecha_prestamo)
LIBROS (cod_libro, editorial, pais)

Atributo fecha_prestamo, no forma parte de la clave pero suministra información de claves candidatas.

## CLASE 07

- Tercera Forma Normal

Se dirá que una relación está en tercera forma normal cuando está en segunda forma normal y no existe dependencia transitiva[10] entre los atributos que no forman parte de la clave.

[10] En otras palabras, Sean A, B y C tres atributos (o grupos de atributos) de una relación. Si B depende funcionalmente de A y C de B, pero A no depende funcionalmente de B se dice que C depende transitivamente de A. La dependencia entre rut-alumno y nombre-curso es transitiva, a través de curso. Intuitivamente se interpreta como que nombre-curso es una información de rut-alumno, pero indirectamente, ya que es una información de codigo-curso y esta a su vez de curso.

Una relación está en 3FN si además de estar en 2FN, los atributos que no forman parte de ninguna clave candidata facilitan información sólo acerca de las claves y no acerca de otros atributos.

Cada atributo, no clave, es dependiente no transitivamente de la clave primaria.

Siguiendo el ejemplo anterior se tiene lo siguiente:

PRESTAMOS en 3FN sería:

PRESTAMOS (num_socio, nombre_socio, cod_libro, fecha_prestamo)
LIBROS (cod_libro, editorial)
EDITORIALES (editorial, pais)

País facilita información acerca de editorial, es por eso que se separa de la relación LIBROS para quedar en 3FN.

Ejemplo

Se tiene la siguiente relación:

FACTURA(nº_factura, fecha_factura, rut_cliente, nombre_cliente, dirección_cliente, nº_vendedor, nombre_vendedor, subtotal, iva, total)

Esta relación se encuentra en 1FN para que quede en segunda y tercera forma normal las relaciones deben quedar de la siguiente forma.

2 FN:

FACTURA (n°_factura, fecha_factura, n°_vendedor, nombre_vendedor, rut_vendedor, rut_cliente, subtotal, IVA, Total)

CLIENTE (rut_cliente, nombre_cliente, dirección_cliente

3 FN:

CLIENTE(rut_cliente, nombre_cliente, dirección_cliente
VENDEDOR(nº_vendedor, nombre_vendedor)
FACTURA(nº_factura, fecha_factura, subtotal, IVA, total)

Como se puede observar se llega a la instancia que las relaciones no se encuentren en dependencias transitivas, es decir, que no puedan descomponerse más.

Existe otra forma normal que permite encontrar la existencia de nuevas relaciones aún estando en su 3FN.

- Forma Normal de Boyce-Cood (FNBC).

Una relación está en FNBC si lo está en 3FN y si además el conocimiento de las claves permite averiguar todas las relaciones existentes entre los datos de la relación. Las claves candidatas deben ser los únicos descriptores sobre los que se facilita información por cualquier otro atributo.

Cada determinante debe ser una clave candidata, siendo un determinante, aquel atributo con el cual otro atributo tiene dependencia funcional total.

Se eliminan las dependencias multivaluadas[11] que no sean funcionales.

En la relación PRESTAMOS nombre_socio y num_socio se repiten por cada libro y préstamo. Nombre_socio se refiere a num_socio y viceversa. (Ninguno de ellos es clave aunque formen parte de ella).

Pasar a FNBC:

SOCIOS (num_socio, nombre_socio)
PRESTAMOS2 (num_socio, cod_libro, fecha_prestamo)

Esquema de 3FN a FNBC sería:

LIBROS1(cod_libro, editorial)
EDITORIALES (editorial, pais)
SOCIOS (num_socio, nombre_socio)
PRESTAMOS2 (num_socio, cod_libro, fecha_prestamo)

Por último se incluirá una pequeña guía de cómo utilizar la teoría de la normalización dentro del diseño lógico. Supóngase que E/R es el esquema relacional obtenido de la transformación del diagrama Entidad/Relación. Estos puntos son una orientación de esta utilización:

[11] Dependencias multivaluadas son una generalización de las dependencias funcionales. En éstas un conjunto de valores del implicado (atributo que depende de otro llamado implicante), son determinados por un implicante. Esta situación aparece cuando existen grupos repetitivos.

a) Comprobar que todas las relaciones incluidas en E/R están en 1FN. Las únicas relaciones que podrían no estarlo son aquellas que provienen de objetos del diagrama Entidad - Relación (entidades o relaciones) que poseían algún atributo estructurado o multivaluado. Transforma estas relaciones a un conjunto de relaciones que estén en 1FN.

Se denota a este nuevo esquema relacional como Entidad - Relación (1FN).

b) Comprobar que todas las relaciones incluidas en E/R (1FN) están en 2FN. Para este punto no se considerarán las claves alternativas, esto es, se supondrá que todas las relaciones sólo poseen una clave candidata. De esta forma los problemas asociados a la definición de atributo no clave desaparecen, siendo más sencilla la aplicación de la definición 2FN. Si alguna relación no está en 2FN descomponerla en un conjunto de relaciones que estén en 3FN. Se denota a este nuevo esquema relacional E/R (2FN).

c) Comprobar que todas las relaciones incluidas en E/R (2FN) están en 3FN. Generalmente, debido a la transformación del diagrama Entidad - Relación al modelo Relacional, las relaciones del E/R (2FN) estarán en 3FN. Es posible que alguna relación no lo esté porque se incluya posteriormente alguna dependencia funcional entre un par de atributos no clave. Transformar estas relaciones a un conjunto de relaciones en 3FN.

d) Comprobar que todas las relaciones incluidas en E/R (3FN) están en FNBC. Para realizar esta tarea se han de considerar las claves candidatas. La presencia de claves candidatas y la posibilidad de permitir determinar el grado de semejanza que presentan 2 o más bases de datos entre ellas es bastante remota, por lo que, generalmente, cada una de las relaciones del esquema relacional E/R (3FN) estará también en FNBC. De no ser así, aplicar la descomposición vista y obtener para la relación que no esté en FNBC un conjunto de relaciones que sí lo estén.

Como punto final de este apartado es interesante recordar que el objetivo final del diseño lógico, en cuanto al diseño de los aspectos estáticos, es obtener un esquema relacional en el cual cada una de las relaciones se encuentra en FNBC. Para ello, cada una de las relaciones incluidas en el esquema relacional obtenido, aplicando las reglas de la transformación del diagrama Entidad – Relación, será analizada para comprobar que cumple esta propiedad. De no ser así se descompondrá en las relaciones adecuadas.

Por otra parte las posibles restricciones de integridad incluidas en el esquema conceptual deberán ser traducidas a expresiones equivalentes del cálculo relacional de tuplas o dominios utilizando como esquema relacional el obtenido después de la aplicación de la normalización.

Realice ejercicios nº 16 al nº 22

# CLASE 08

## 6. DIAGRAMA DE DEPENDENCIA FUNCIONAL (DDF)

El modelo relacional se afirma en el concepto de dependencia funcional, en particular el proceso de normalización[12] por las cuales se busca rediseñar las relaciones de forma tal que sus estructuras sean las óptimas desde el punto de vista lógico. Es decir, que a través del proceso de normalización con el concepto de dependencia funcional se pueda eliminar la redundancia de datos y los problemas que puedan surgir al momento de realizar una actualización, eliminación o modificación mientras se trabaja en las relaciones y sus asociaciones.

La obtención de un Esquema Relacional (ER) ha sido a partir de la descripción de un problema, ya sea, de forma directa, o vía modelo entidad-relación que posteriormente es transformado a un modelo relacional.

Lo dicho anteriormente supone que desde la descripción del problema se puede deducir sin mayores inconvenientes las relaciones y las asociaciones implicadas, así como sus atributos y dominios.

Este enfoque en general de carácter Top – Down[13], implica identificar primero las relaciones y sus asociaciones para luego especificar sus atributos así como sus respectivos dominios.

Otro enfoque totalmente inverso a lo descrito anteriormente es de carácter botton-up[14], que propone identificar los atributos y luego las relaciones y sus asociaciones. Esto presume que en una descripción de una problemática dada resultará más fácil encontrar eventuales atributos, para luego determinar las relaciones en que se encuentran en base al concepto de dependencia funcional.

Un ejemplo válido es cuando se tiene unos juegos LEGOS, en donde, los atributos serían las piezas de los LEGOS que se deben ensamblar y encajar. Ensamblar implica estructurar las relaciones que calcen perfecto para entender de mejor manera la información que entrega y encajar implica establecer las asociaciones entre las relaciones.

Tanto el montaje y ajustamiento es realizado en base a los diagramas de dependencia funcional.

Una definición para un diagrama de dependencia funcional (DDF) sería que es un conjunto de nodos que representan atributos y en el que los arcos representan dependencias funcionales.

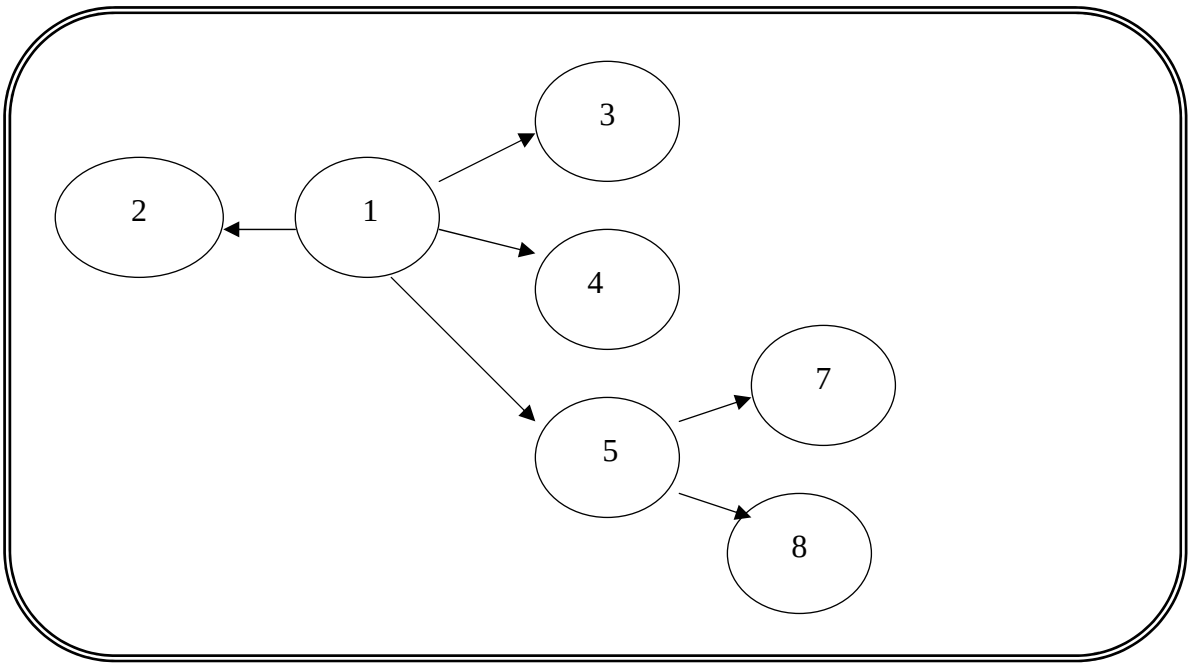
---

[12] Normalización: es un proceso destinado a eliminar problemas, anomalías que pueden salir de las estructuras de las tablas. Este concepto nace en el modelo relacional.
[13] Top dow, es decir, se parte de l más general hasta llegar a lo más específico
[14] Botton-up: Se parte de lo mas especifico a lo mas general

A continuación, se observa que existen diagramas funcionales de los atributos:

- 2, 3, 4 y 5, en relación del atributo 1; y
- 7, 8 en relación del atributo 5

El diagrama de dependencia funcional que representan estos diagramas funcionales es el que a continuación se detalla:




En base a las dependencias funcionales dadas, se puede desprender una entidad relación constituida por las siguientes relaciones:

R1 (1, 2, 3, 4, 5)

R2 (5, 7, 8)

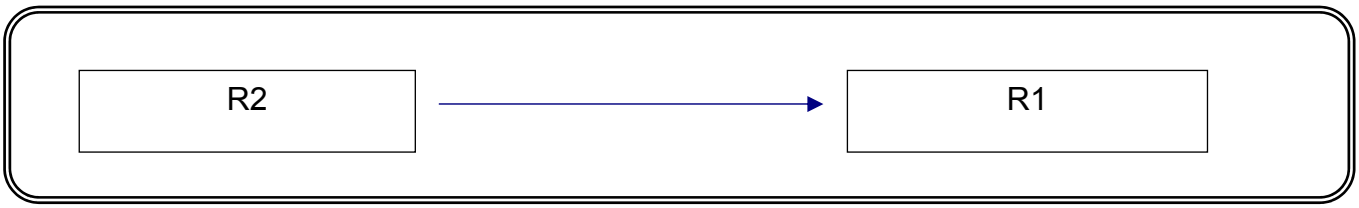
Como se puede observar las relaciones fueron conformadas a partir de las dependencias funcionales mostradas de un mismo atributo. En el diagrama se puede demostrar que los atributos 2, 3, 4, 5 dependen funcionalmente de un mismo atributo que es el atributo 1. Mientras que los atributos 7, 8 dependen funcionalmente de 5.

Otra forma de representar las dependencias funcionales que se manifiestan sería la siguiente:

1 → 2, 3, 4, 5

5 → 7, 8

Ahora si se piensa en el concepto de clave foránea[15], se puede deducir que entre las relaciones R1 y R2 existe la siguiente asociación:




A continuación, se analiza un ejemplo más concreto llevado a la vida cotidiana.

Realice ejercicio nº 23

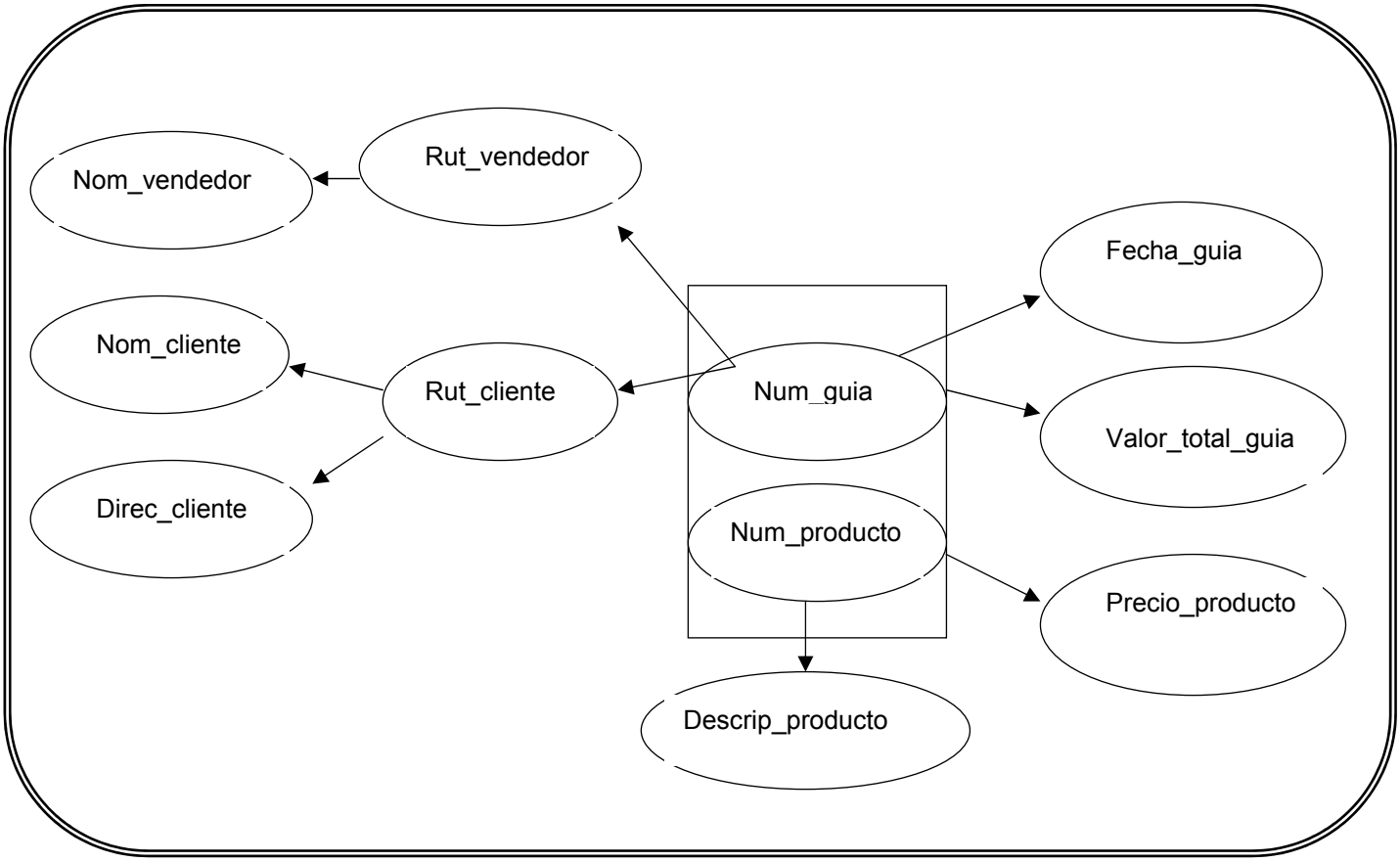
Ejemplo

1) Se tienen los siguientes datos:

a) num_guia,
b) nombre_vendedor
c) descripción_producto
d) num_producto
e) rut_vendedor
f) rut_cliente
g) nombre_cliente
h) fecha_guía
i) precio_producto
j) valor_total_guía
k) dirección_cliente.

Con los datos que se tienen se puede deducir el diagrama de dependencia funcional siguiente:

[15] Clave Foránea: es simplemente una columna que se refiere a la clave primaria de otra tabla.




Se puede observar que el diagrama de dependencia funcional muestra que del atributo num_guia dependen funcionalmente los atributos rut_cliente, rut_vendedor, fecha_guia, valor_total_guia, precio_producto. Del atributo num_producto dependen funcionalmente precio_producto, descripción_producto.

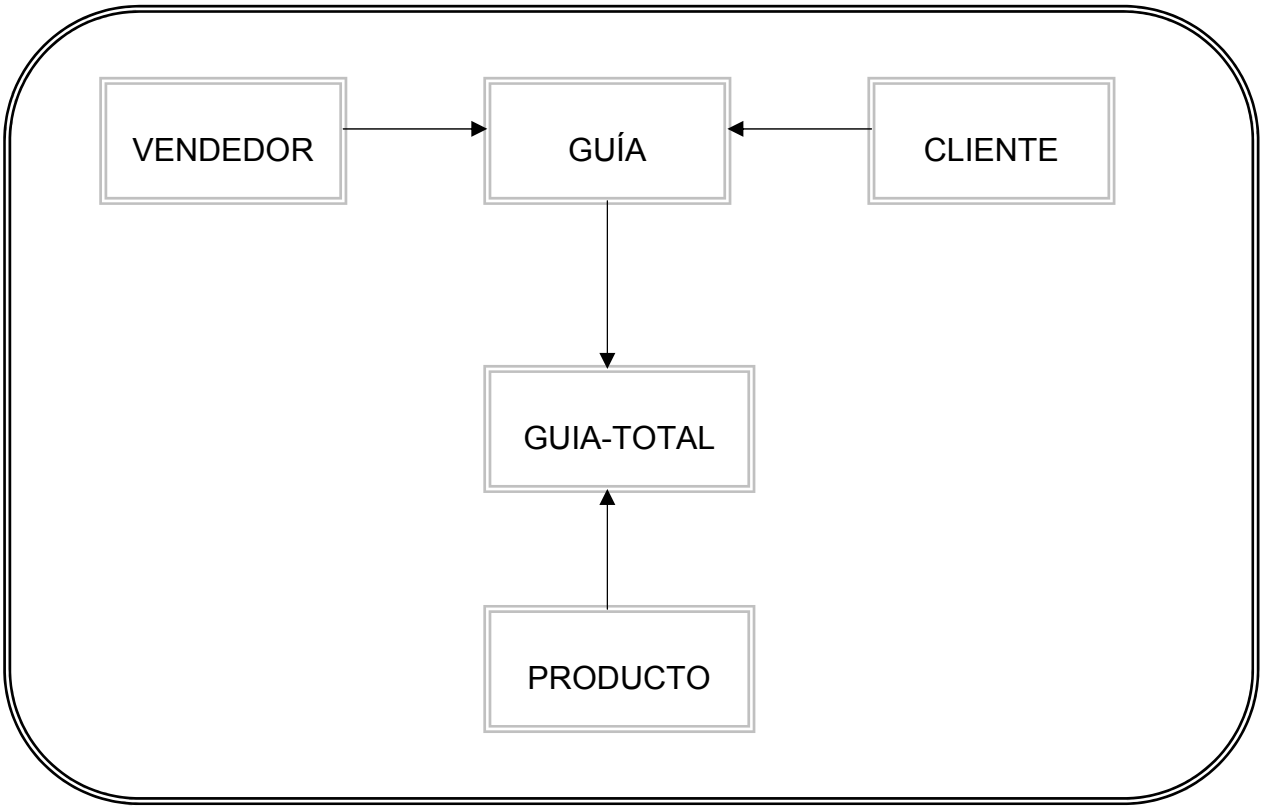
Del atributo rut_cliente dependen funcionalmente nombre_cliente y dirección_cliente y finalmente del atributo rut_vendedor depende funcionalmente el atributo nombre_vendedor.

En consecuencia las relaciones que se manifiestan serían:

GUIA (numero_guia, fecha_guia, valor_total_guia, rut_vendedor, rut_cliente)
VENDEDOR (rut_vendedor, nombre_vendedor)
CLIENTE( rut_cliente, nombre_cliente, dirección_cliente)
PRODUCTO (numero_producto, descripción_producto, precio_producto)
GUIA -TOTAL (numero_guia, numero_producto)

Finalmente el diagrama relacional asociado a estas relaciones quedaría del siguiente modo:

Ejemplo




# CLASE 09

## 7. ANÁLISIS DE DOCUMENTOS

Entre las fuentes de modelamiento de datos más útiles se encuentran los documentos o formularios existentes en las organizaciones y que se aplican para hacer efectivos los procesos en que se encuentran envueltos.

Un documento tiene una estructura que viene dada por un encabezado, uno o más cuerpos, y un pie del documento.

Los casos que se pueden presentar son los siguientes:

a) Formulario sin cuerpo: corresponde al caso en que no se incluyen líneas de detalle y por tanto no contiene pie. El cheque es un ejemplo en el que todo su contenido constituye el encabezado del documento.

Ejemplo

| | |
|---|---|
| Nº Pasaje | |
| Fecha Emisión | Lugar Emisión |
| | |
| Origen | Destino |
| | |
| Fecha Salida | Hora Salida |
| Valor Pasaje | |

Este ejemplo corresponde a un pasaje y toda su información también pertenece a un encabezado

b) Formulario con un cuerpo: corresponde al caso que incluye líneas de detalle repetitivas bajo una misma estructura, pudiendo incluir o no un pie. Las boletas son un ejemplo de este tipo de formularios.

Ejemplo

| CARTOLA DE PAGOS | | | | |
|---|---|---|---|---|
| # Cartola | Fecha | Total Compras | Total Pagos | Monto Deuda |
| Rut Cliente | Nombre Cliente | Dirección Cliente | Fono Cliente | |
| # Pago | Fecha Pago | Monto Pago | Descripción Pago | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Este es un ejemplo de una cartola de pagos

Formulario con más de un cuerpo: corresponde al caso que incluye líneas de detalle repetitivas con más de una estructura, pudiendo incluir o no un pie. Las cartolas bancarias pueden ser un ejemplo de este tipo de formularios cuando incluyen los abonos bajo una estructura y los cargos bajo otra estructura.

Ejemplo

| | | | | |
|---|---|---|---|---|
| Nº Factura<br>Fecha | | | | |
| | | | | |
| Rut Cliente | Dirección Cliente | Fono Cliente | Giro Cliente | |
| Cod Artículo | Descripción Artículo | Precio Unitario | Cantidad | Valor |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | Subtotal | |
| | | | IVA | |
| | | | Total | |
| Fecha | | | | |
| Nº Guia | | | | |

En este tipo de formulario llamado factura se observa la existencia de 2 cuerpos. El primero hace referencia a los artículos por los cuales se factura, y el segundo a las guías de despacho que respaldan las compras efectuadas y que dan origen a la factura.

Para poder entender un poco más el análisis del documento, se estudiará un ejemplo en los que se construye un ER a partir de formularios o documentos. Interesa destacar que muchas veces la mejor manera de modelar una realidad dada es a partir de los documentos que actualmente manejan las empresas.

Ello por causa de la escasa disponibilidad de tiempo que suelen tener los ejecutivos responsables de los procesos o de las unidades funcionales en que se estructura la empresa. Lo mencionado anteriormente es válido al menos para obtener una primera versión del modelo.

## Ejemplo

1.- Se tiene el siguiente ejemplo del que se pide construir el ER asociado de forma tal que las relaciones se encuentren normalizadas.

| ESTADO DE AVANCE DE PAGOS | | | | |
|---|---|---|---|---|
| #Estado de Avance | Fecha | Total compras | Total Pagos | Monto deuda |
| Rut Cliente | Nombre Cliente | Dirección Cliente | Fono Cliente | |
| # Pago | Fecha Pago | Monto Pago | Descripción Pago | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

La relación que se rescata del documento es:

ESTADO_AVANCE (#estado, fecha, total_compras, total_pago, deuda, rut_cliente, nombre_cliente, dirección_cliente, fono_cliente (#pago, fecha_pago, monto_pago, descripción_pago))

Esta relación no se encuentra en su 1 forma normal, porque para cada # estado los atributos(#pago, fecha_pago, monto_pago, descripción_pago) son multivaluados. Por lo tanto de debe normalizar.

La relación quedaría:

1FN 2FN: ESTADO_AVANCE (#estado, fecha, total_compras, total_pago, deuda, rut_cliente, nombre_cliente, dirección_cliente, fono_cliente.

DETERMINAR_ESTADO_AVANCE (#pago, fecha_pago, monto_pago, descripción_pago)

Ambas relaciones se encuentran no sólo en su 1 FN, si no que también en su 2FN, porque todos sus atributos tienen dependencia funcional completa[16] respecto de sus claves. Sin embargo todavía falta que se cumpla la tercera forma normal debido a que existe todavía dependencias funcionales de los atributos nombre_cliente, dirección_cliente y fono_cliente con respecto al rut_cliente.

La relación quedaría:

3FN:

ESTADO_AVANCE (#estado, fecha, total_compras, total_pago, deuda)

DETERMINAR_ESTADO_AVANCE (#pago, fecha_pago, monto_pago, descripción_pago

CLIENTE (rut_cliente, nombre_cliente, dirección_cliente, fono_cliente)

El diagrama relacional sería:


Realice ejercicio nº 24

[16] Dependencia funcional completa: siendo x e y atributos de una misma relación y depende funcionalmente en forma completa de otro atributo x. En donde y pertenece a x y no que x pertenezca a y. Solo pertenece a un atributo independiente que este atributo tenga otros atributos como subconjuntos de él.

# CLASE 10

## 8. ÁLGEBRA RELACIONAL

El álgebra relacional consiste en un conjunto de operadores de alto nivel que operan sobre relaciones con el fin de formular consultas a la base de datos. Cada uno de estos operadores toma una o dos relaciones como entrada y produce una nueva relación como salida (propiedad de clausura).

Codd definió un conjunto de 8 operadores, los que se describen a continuación:

### a) Selección.

La selección también denominada restricción, opera sobre una relación R y da como resultado otra relación, cuyas tuplas son las tuplas de R que satisfacen la condición especificada. Selecciona la información que se pide mediante la consulta que se especifica.

La mencionada condición corresponde a una comparación en donde aparece al menos un tributo de R.

El operador de selección utiliza la letra griega (δ) para denotar la selección aplicado a la relación R, el predicado aparece como subíndice de (δ), la relación del argumento, vale decir, el nombre de la tabla que se extraerá la información, en este caso se da entre paréntesis.

Por ejemplo

Para seleccionar las tuplas de una relación PRESTAMO en que la sucursal es "Talca" sería:

(δ) nombre_sucursal = "Talca" (prestamo)

### b) Proyección

La proyección opera sobre una relación R y da como resultado otra relación que contiene un subconjunto vertical de R, extrayendo los valores de los atributos especificados y eliminando duplicados.

Este operador extrae los atributos especificados de una relación dada, se utiliza la letra griega (π) que indica la operación que se quiere realizar. Además es parte de la sintaxis de la operación.

Por ejemplo

Se quiere una lista de todos los números de préstamo y los importes pero no requiere el nombre de las sucursales.

(π) numero_prestamo, importe (prestamo)

c) Unión

Si se considera la unión como aquella efectuada entre dos relaciones R y S, donde P y Q son tuplas, corresponde a otra relación que tiene cuando mucho P + Q tuplas siendo éstas las tuplas que se encuentran en R o en S o en ambas relaciones a la vez. Para poder realizar esta operación, R y S deben ser compatibles para la unión.

Entiéndase por compatibles para la unión cuando dos relaciones tienen la misma cabecera, es decir, si tienen el mismo número de atributos y éstos se encuentran definidos sobre los mismos dominios.

Construye una relación formada por todas las tuplas que aparecen en cualquiera de las dos relaciones especificadas, también se utiliza la letra griega (π), para mostrar la operación a realizar que en este caso sería la unión. Ahora para que haya una operación de Unión se debe cumplir lo siguiente:

1.- Las relaciones deben tener la misma cantidad de atributos
2.- Los dominios de los atributos deben ser iguales

Por ejemplo

Averiguar el nombre de todos los clientes del banco que tienen una cuenta y un préstamo.

(π) nombre_cliente (prestatario) U (π) nombre_cliente (impositor)

d) Producto

El operador producto consiste en multiplicar dos relaciones, definiendo una nueva relación que tiene todos los pares posibles de tuplas de las dos relaciones. Si la relación R tiene P tuplas y N atributos y la relación S tiene Q tuplas y M atributos, la relación resultado tendrá P x Q tuplas y N + M atributos. Puesto que es probable que existan atributos con el mismo nombre en las dos relaciones, el nombre de la relación se antepondrá al del atributo, en este caso para que los nombres de los atributos sigan siendo únicos en la relación resultado.

Se denota con la aspa (x).

Por ejemplo

Se quiere averiguar por todos los nombres de los clientes que tienen concedido un préstamo en la sucursal de Talca.

(x) nombre_sucursal = “Talca” (prestatario x prestamo)

e) Intersección

Este operador obtiene como resultado una relación que contiene las tuplas de R que también se encuentran en S. Para realizar esta operación, R y S deben ser compatibles para la unión.

En definitiva construye una relación formada por todas aquellas tuplas que aparecen en las dos relaciones especificadas, utilizando la letra griega (π).

Por ejemplo

Se desea averiguar por todos los clientes que tienen un préstamo concedido y una cuenta abierta.

(π) nombre_cliente (prestatario) ∩ (π) nombre_cliente (impositor)

Dadas dos relaciones, con esquemas compatibles R1 y R2, la intersección de ambas, se denota de la siguiente manera:

R1 ∩ R2

f) Diferencia

Este operador construye una relación formada por todas las tuplas de la primera relación que no aparezcan en la segunda de las dos relaciones especificadas. Se utiliza la letra griega (π).

Por ejemplo

Buscar todos los clientes que tienen una cuenta abierta pero que no tienen concebido ningún préstamo.

(π) nombre_cliente (prestatario) - (π) nombre_cliente (impositor)

Dadas dos relaciones, con esquemas compatibles R1 y R2, la intersección de ambas, se denota por:

R1 - R2

## g) Reunión

En este caso, a partir de dos relaciones especificas, se construye una relación que contiene todas las posibles combinaciones de tuplas, una de cada una de las dos relaciones, tales que las dos tuplas participantes en una combinación dada satisfagan alguna condición especificada. Se utiliza la letra griega (π).

Por ejemplo

Averiguar los nombres de todos los clientes que tienen un préstamo en el banco y averiguar su importe o saldo.

(π) nombre_cliente, prestamo, numero_prestamo (prestatario x prestamo)

## h) División

Éste toma dos relaciones, una binaria[17] y otra unaria[18], y construye una relación formada por todos los valores de un atributo de la relación binaria que concuerdan (en el otro atributo) con todos los valores en la relación unaria.

Por ejemplo

Encontrar a todos los clientes que aparecen con el nombre de todas las sucursales = Talca.

(π) nombre_cliente, nombre_sucursal ( importe x cuenta)

DIVIDIDO

(π) nombre_sucursal, ((δ) ciudad_sucursal = Talca ( sucursal)

Este operador toma dos relaciones y construye una relación consistente de todos los atributos de la primera relación que no están en la segunda relación.

[17] Binaria: Porque se opera con relaciones pares.
[18] Unaria : porque operan sobre una sola relación.

Creará una relación con todos los clientes que no tengan la ciudad sucursal = Talca.

La división se puede expresar en función de la proyección, del producto cartesiano y de la diferencia de la siguiente forma:

R1 : R2 = π C(R1) - πC (R2 x πC (R1) - R1) siendo C el conjunto de atributos de A menos los de B.

Realice ejercicio nº 25 al n° 26

# CLASE 11

## 9. EL LENGUAJE DE LAS BASES DE DATOS RELACIONALES

.

SQL es una herramienta para organizar, gestionar y recuperar datos almacenados en una base de datos informática. El nombre de "SQL" es una abreviación de Structured Query Language (Lenguaje Estructurado de Consultas).

Como su nombre indica, SQL es un lenguaje informático que se utiliza para interactuar con un tipo de bases de datos, llamado base de datos relacional. SQL es un lenguaje de bases de datos relacionales, y ha llegado a ser popular junto con el modelo relacional de base de datos.

La estructura tabular fila/columna de las bases de datos relacionales es intuitiva para los usuarios, manteniendo el lenguaje SQL sencillo y fácil de entender.

SQL permite la definición de datos, la recuperación de datos, la manipulación de datos, y el control de acceso, así como compartir la información y la integridad de los datos.

- Características y ventajas de SQL

A continuación, se detallan las características y ventajas del lenguaje SQL.

- Es un lenguaje de definición de datos que proporciona ordenes para la definición de esquemas de relación, borrado de relaciones, creación de índices y modificación de esquemas de relación.

- Incluye un lenguaje de consultas, basado tanto en el álgebra relacional como en el calculo relacional de tuplas. Incluye también ordenes para insertar, borrar y modificar tuplas de la base de datos.

- SQL se diseñó de manera que no se use ningún lenguaje de programación.
- Incluye órdenes para la especificación del comienzo y final de las transacciones.

Aunque el lenguaje SQL se considere un lenguaje de consulta, contienen muchas otras capacidades además de las consultas en bases de datos, las cuales se estudiarán con posterioridad.

## 9.1. Estructura Básica

Una base relacional consiste en un conjunto de relaciones a cada una de las cuales se le asigna un nombre único.

La estructura básica de una expresión SQL consiste en tres cláusulas: select, from, where.

a) La cláusula Select corresponde a la operación proyección del Álgebra Relacional. Se usa para listar atributos deseados del resultado de una consulta.

b) La cláusula From corresponde a la operación Producto Cartesiano del Álgebra Relacional. Lista las relaciones que deben ser analizadas en la evaluación de la expresión.

c) La cláusula Where, corresponde a la operación Selección del Álgebra Relacional Es un predicado que engloba a los atributos de las relaciones que aparecen en la cláusula From .

A continuación, se analizarán cada una de las cláusulas con detalle.

- SELECT

El resultado de una consulta SQL es una relación, si se tiene una consulta simple como por ejemplo: obtener los números de todas las sucursales en la relación préstamo:

```
SELECT nombre-sucursal
FROM préstamo
```

El resultado es una relación consistente en el único atributo nombre-sucursal.

Los lenguajes formales de consulta están basados en la noción matemática de que una relación es un conjunto, de este modo nunca aparecen tuplas duplicadas en las relaciones. En la práctica la eliminación de duplicados consume tiempo, sin embargo en SQL, permite duplicados en las relaciones.

Así la consulta anterior listará cada nombre-sucursal una vez por cada tupla en la que aparece en la relación préstamo.

En aquellos casos en que se quiera forzar la eliminación de duplicados, se insertará la palabra clave distinct después de select. Por lo tanto la consulta anterior se puede escribir del siguiente modo.

Si se desea eliminar los duplicados sería:

```
Select distinct nombre-sucursal
From préstamo
```

Ahora cabe destacar que SQL, permite usar la palabra clave all para especificar explícitamente que no se eliminan los duplicados, se representa de la siguiente manera:

```
Select all nombre-sucursal
From préstamo
```

El símbolo asterisco (*) se puede usar para denotar todos los atributos. Así el uso de préstamo.* en la cláusula select indicaría que todos los atributos de préstamo serían seleccionados.

Una cláusula select de la forma select * indica que se deben seleccionar todos los atributos de todas las relaciones que aparecen en la cláusula from.

La cláusula select puede también tener expresiones aritméticas que contengan los operadores +, -, * y / operando sobre constantes o atributos de la tuplas.

## CLASE 12

Ejemplo:

```
Select nombre-sucursal, numero-préstamo, importe * 100
From préstamo
```

El ejemplo indica que el resultado será una relación que es igual que la relación préstamo, salvo que el atributo importe está multiplicado por 100.

- WHERE

Se analiza el siguiente ejemplo: “ tenemos la consulta: obtener todos los números de préstamo para préstamos hechos en la sucursal con nombre Talca, en los importes que sean superior a $200.000”.

En SQL se escribe de la siguiente manera:

```
Select número-prestamo
From préstamo
Where nombre-sucursal = “Talca”  and  importe > 200.000
```

SQL usa los conectores lógicos and, or y not en la cláusula where. Los operandos de los conectores lógicos pueden ser expresiones que contengan los operadores de comparación como lo son : <, >, <=, >=, =. SQL permite usar los operadores de comparación para comparar expresiones aritméticas.

SQL incluye un operador de comparación between para simplificar las cláusulas where que especifica que un valor sea menor o igual que un valor y mayor o igual que otro valor.

Ejemplo

```
Select numero-préstamo
From préstamo
Where importe between  600.000 and  3.000.000
```

En lugar de:

```
Select numero-prestamo
From prestamo
Where importe <= 3.000.000  and importe >= 600.000
```

Ejemplo

TABLA CLIENTE

| Rut | Nombre | Sucursal Banco | Sueldo |
|---|---|---|---|
| 8.256.963-7 | Juan Pérez | Talca | 500.000 |
| 11.951.788-k | María José Soto | San Javier | 600.000 |
| 6.896.216-1 | José Rodriguez | Talca | 300.000 |

Obtener todos los nombres de aquellos clientes cuya sucursal bancaria sea Talca y el sueldo mayor de 400.000

```
Select nombre-cliente
From Cliente
Where  sucursal= "Talca" and sueldo > 400.000
```

El resultado es:

Juan Pérez

- FROM

Esta cláusula define por sí misma un producto cartesiano de las relaciones que aparecen en la cláusula. Se tiene por ejemplo: "Obtener los nombres y números de préstamo de todos los clientes que tienen un préstamo en el Banco".

Ejemplo

```
Select distinct nombre-cliente, prestatario.número-préstamo
From prestatario, préstamo
Where prestatario.número-préstamo = préstamo.número-préstamo
```

Se observa que en SQL se usa la notación nombre-relación.nombre-atributo, para evitar ambigüedad en los casos en que un atributo aparece en el esquema de más de una relación.

También se podría haber escrito prestatario.nombre-cliente, en lugar de nombre-cliente en la cláusula select. Pero como el atributo nombre-cliente aparece sólo en una de las relaciones de la cláusula from, no existe ambigüedad al escribir nombre-cliente.

Otro ejemplo podría ser que además de lo anterior se pida que los clientes posean un préstamo en la sucursal de "Talca", "Obtener los nombres y números de préstamos de todos los clientes que tienen un préstamo en la sucursal de Talca.

Para escribir esta consulta será necesario establecer dos ligaduras en la cláusula where, relacionadas con el conectivo and.

Ejemplo

```
Select  distinct nombre-cliente, prestatario.numero-préstamo
From  pretatario, préstamo
Where  prestatario.numero-prestamo  =  prestamo.numero-prestamo  and
nombre-sucursal = "Talca"
```

## 10. VALORES NULOS EN LENGUAJE SQL

SQL permite el uso de valores nulos para indicar la ausencia de información sobre el valor de un atributo.

Se puede usar la palabra clave null para comprobar si un valor es nulo. Así para encontrar todos los números de préstamo que aparecen en la relación préstamo con valores nulos para importe sería:

```
Select  numero-préstamo
 From  préstamo
 Where  importe  is  null
```

El predicado is not null pregunta por la ausencia de un valor nulo.

El uso de un valor nulo en las operaciones aritméticas y de comparación causa varias complicaciones. El resultado de una expresión aritmética (+, -, *, / ) es nulo si cualquiera de los valores de entrada es nulo.

El resultado de cualquier comparación que involucre un valor nulo se puede considerar falso.

La existencia de valores nulos también complica el procesamiento de los operadores de agregación. Por ejemplo supongamos que algunas tuplas en la relación préstamo tienen valor nulo para el atributo importe. Se considerará la consulta que calcule el total de todas las cantidades prestadas:

Ejemplo

```
Select sum (importe)
From préstamo
```

Los valores que van a ser sumados en el ejemplo anterior incluyen valores nulos, puesto que algunas tuplas tienen valor nulo para el atributo importe. En lugar de decir que la suma total es nula, la norma SQL establece que el operador sum debería ignorar os valores nulos de su entrada.

En general las funciones de agregación tratan los valores nulos según la regla siguiente:

"Todas las funciones de agregación excepto count ignoran los valores nulos de la colección de valores de datos de entrada".

Realice ejercicios n° 27 al nº 30