CURSO:
BASES DE DATOS
Unidad I: Modelamiento de datos.

Clase 1: Introducción a las bases de datos.

Profesor: Diego Miranda

Unidades y Objetivos del Curso:

- UNIDAD I: Modelamiento de datos
- UNIDAD II: Implementación de una base de datos
- UNIDAD III: Interactuando con una base de datos

Calendario de evaluaciones:

- A definir…

Para comenzar…

- Abstracción: operación mental destinada a aislar conceptualmente las propiedades y características de un objeto, sea este material o inmaterial
- Dato: representación simbólica de un atributo o variable cualitativa o cuantitativa
- Atributo: especificación que define una propiedad o característica de un objeto
- Entidad: Es la agrupación de un conjunto de atributos en un contexto
- Valor: corresponde al dato asociado a describir particularmente un atributo de un objeto
- Instancia: conjunto o espacio de valores que asumen los atributos de un objeto en un momento particular
- Objeto: es cualquier “cosa” desde el punto de vista ontológico

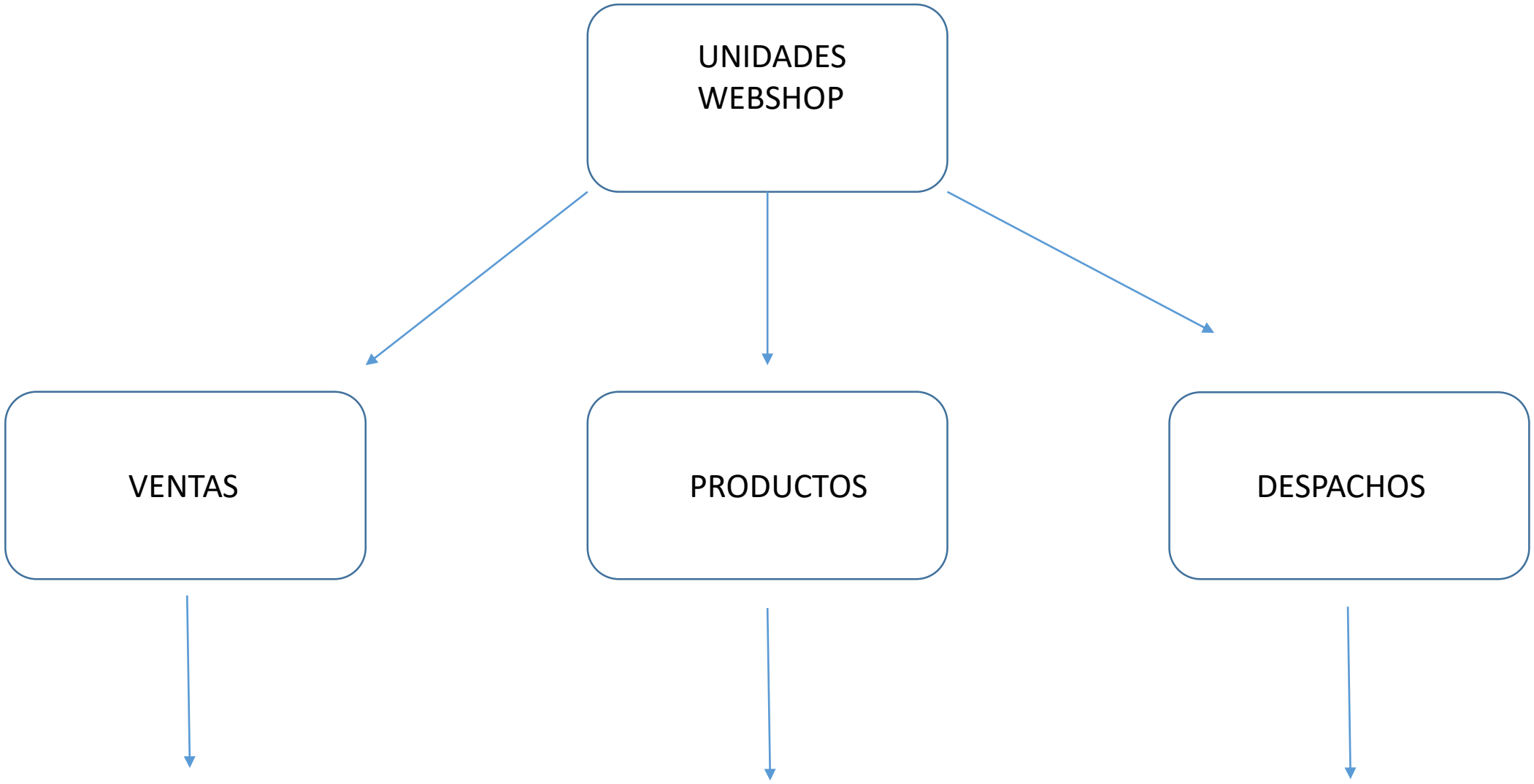
INTRODUCCIÓN

Hoy en día todas las empresas usan datos de una forma u otra. Típicamente, muchos de estos datos vienen en formatos como Excel, Word, o archivos de texto. Aunque uno puede guardar sus datos de esta manera, con el crecimiento de la organización, este tipo de solución va a presentar varios problemas. Por ejemplo, se debe asegurar que solo los gerentes del área de finanzas pueden revisar los sueldos de los empleados, o que los datos del mismo empleado no aparecen dos veces en la lista de liquidaciones de sueldos. Para evitar este tipo de problemas, la mayoría de las empresas hoy en día utilizan un **sistema de bases de datos**.

Imaginemos que Juan…

tiene una tienda on-line que vende productos tecnológicos llamada WebShop. Originalmente Juan vendía solo cinco tipos de cámaras fotográficas, y manejaba todas las ventas por correo electrónico. Después de esto, Juan automatizó el proceso implementando una interfaz Web en Python, donde los clientes podían hacer su pedido, y a Juan le llegaba una notificación sobre la compra.

Como la demanda para sus productos crecía, Juan tuvo que contratar más gente para manejar diferentes procesos en su empresa. En este momento la empresa WebShop se estructuró en 3 áreas…


UNIDADES WEBSHOP
VENTAS
PRODUCTOS
DESPACHOS
Encargada de las ventas de los productos
Controlar el stock de prod. disponibles
Mantiene el registro de clientes y realiza el despacho de productos

¿Cómo se ve un día en la empresa de Juan?

Primero, un cliente hace su pedido en la página Web del WebShop. El pedido es informado a la unidad de productos para revisar si existe stock suficiente para satisfacerlo. Una vez verificada, esta información se envía a la unidad de ventas que registra si el pago del producto se realizó con éxito. Cuando se realiza el pago, se envía la información a la unidad de despachos, para realizar el envío del producto. Obviamente, todas las unidades involucradas en el proceso tienen que guardar ciertos datos sobre la compra. En lo que sigue, vamos a explicar cómo las unidades del WebShop guardan sus datos

Distintos Formatos de Datos

Para permitir el funcionamiento de WebShop, cada unidad de negocios tiene que manejar ciertos datos. Por ejemplo, la **unidad de productos** mantiene una plantilla Excel llamada Productos, que contiene información sobre cada producto que se vende en WebShop. Cada fila en esta plantilla contiene el código, nombre, precio y el stock disponible de cada producto. En la siguiente imagen se muestra una parte de esta planilla:

Código	Nombre	Precio	Stock
123	Sony AC2020	$150,000	2
127	Panasonic KS12	$250,000	0
143	GoPro Hero 4	$199,999	12
229	GoPro Hero 5	$198,999	2
301	Apple Mouse	$1,000,000	5

Por otro lado, la **unidad de ventas** maneja las ventas en el archivo de Excel llamado Ventas. Aquí se guarda la información sobre cuál usuario compró qué producto, y si el pago fue procesado o está pendiente. Una imagen del archivo Ventas se aprecia a continuación:

RUT	Código	Status
15.378.AAA - K	123	Pagado
18.268.BBB - 7	143	Pagado
18.268.BBB - 7	301	Pendiente
18.268.BBB - 7	229	Pagado

Finalmente, la **unidad de despachos** guarda un registro de los usuarios y sus direcciones. Para ahorrarse el tiempo de armar un Excel, y para poder copiar y pegar la dirección de despacho directamente desde su registro, la unidad de despachos guarda sus datos en un archivo de texto llamado Usuarios. Este archivo contiene el nombre de la persona, domicilio, comuna etc. Adicionalmente, para permitir personas con el mismo nombre en sus registros, la **unidad de despachos** los separa por su RUT. Ejemplo de unos usuarios podemos ver en la imagen abajo

¿Cómo se utilizan los datos?

Ahora que sabemos cómo cada unidad de WebShop guarda sus datos, podemos revisar como estos datos son utilizados cuando un cliente realiza la compra en el sitio. Para esto vamos a considerar el caso de Gonzalo, quién quiere comprar una GoPro Hero 5 en WebShop. Gonzalo navega a la página en internet de la empresa, y después de encontrar el producto GoPro Hero 5, lo agrega a su carrito de compras. Para finalizar su compra realiza el pago y luego ingresa su dirección donde recibirá el producto. En este momento se genera un orden de compra con el monto $198.999. Veamos cómo se reflejan estas acciones en los datos guardados por distintas unidades de WebShop.

- **Primero**, la unidad de productos revisa que exista stock suficiente de GoPro Hero 5, para que el pedido sea clasificado como factible. Luego actualizan su planilla Productos y también la página web. En particular, el stock de GoPro Hero 5 baja de 2 a 1.
- **Segundo**, la unidad de ventas chequea el pago de Gonzalo al revisar la cuenta de la empresa y registra en la planilla Ventas que Gonzalo pagó exitosamente el producto con el código 229. Esta verificación es necesaria porque ante algún problema en el pago el status debe quedar como Pendiente. Si el pedido fue clasificado como infactible por falta de stock se debe reversar el pago. Ahora, la planilla Ventas contiene la siguiente fila:

RUT	Código	Status
18.268.BBB – 7	229	Pagado

- **Tercero**, la unidad de despachos debe registrar la dirección de Gonzalo en el archivo Usuarios. En el caso que Gonzalo ya haya realizado alguna compra en WebShop, puede ocurrir que este dato ya existe. Si la dirección de Gonzalo cambió, esto también se registra en el archivo. Finalmente, la unidad de despachos envía el GoPro a Gonzalo, terminando el proceso.

Problemas organizacionales

La manera en la cual WebShop maneja a sus datos tiene ciertas debilidades que pueden causar varios daños económicos, desde la pérdida de ventas, hasta la necesidad de reorganizar la empresa. Examinemos unos ejemplos de esto.

Ejemplo 1: Integridad de los datos

¿Qué pasa cuando dos personas intentan acceder a un dato al mismo tiempo? Volviendo a la compra de Gonzalo, pensemos cómo esta compra es procesada por la unidad de ventas. Andrea y Jorge trabajan en la unidad de ventas y reciben la misma notificación sobre el pedido de Gonzalo desde la unidad de productos. Jorge actualiza la plantilla Ventas, e inmediatamente avisa a la unidad de despachos para que envíen una GoPro Hero 5 al cliente. Al mismo tiempo Andrea hace lo mismo ya que trabajan en oficinas separadas y no pudieron comunicarse. Como los cambios de Jorge no alcanzaron a ser guardados en la plantilla, Andrea correctamente cambia el estatus de la compra de Gonzalo a “pagado”, y envía un orden de despacho a la unidad de productos. Pensando que se trata de dos ventas distintas, la unidad de productos actualiza el stock del producto GoPro Hero 5 a cero, y con un solo pago, Gonzalo recibe dos unidades del producto. El problema en este caso fue que dos personas estaban utilizando el mismo dato al mismo tiempo, sin conocer las acciones del otro. Errores similares pueden ocurrir cuando dos clientes intentan comprar al mismo producto con solo una unidad en stock.

Ejemplo 2: Integración de los datos

Un problema grande de soluciones hechas a medida para el manejo de datos, como en el caso de WebShop, es que cuando uno quiere cambiar la lógica de los procesos en la empresa, tiene que rediseñar todo desde cero. Por ejemplo, si WebShop hace una adquisición de VegShop, la tienda on-line para compra de vegetales y frutas, las dos empresas van a tener que unir sus datos. ¿Qué pasa si VegShop usa otros formatos para guardar a sus datos? Por ejemplo, si todos los datos del VegShop están en archivos de texto o si VegShop utiliza 1 gran planilla centralizada que incluye los datos del cliente junto a la orden de compra. Similarmente, si la aplicación Web de WebShop está programada en Python y la del VegShop en Java, para crear una interfaz Web común vamos a tener que implementar una de las soluciones de nuevo, incrementando los costos de operación de la empresa.

Ejemplo 3: Eficiencia

Finalmente, manejar datos de manera no automatizada, como en el caso de WebShop, puede ser muy ineficiente. Por ejemplo, para enviar un producto al cliente, la unidad de despachos tiene que esperar confirmación de la unidad de ventas. Si la gente de ventas está tomando una pausa, este paso puede tardar. Incluso, si todo el proceso fuese automatizado, una vez que la cantidad de datos crece mucho (por ejemplo, cuando las plantillas tienen millones de filas), buscar los datos se vuelve muy lento, porque el software utilizado no está especializado para este tipo de tareas.

Cómo solucionamos esos problemas???

SISTEMAS DE BASES DE DATOS (DBSM)

Como podemos ver, implementar soluciones a medida tiene ciertas limitaciones, y puede incurrir costos adicionales, o incluso errores graves en el funcionamiento de una empresa. Afortunadamente, el manejo de datos es uno de los problemas más estudiados en el área de la Computación, y existen soluciones que permiten evitar estos tipos de problemas.

¿Qué es una base de datos, y qué es un sistema de bases de datos?

Para partir, primero vamos a definir formalmente el concepto de bases de datos.

- **Base de datos**: Una base de datos es una colección de los datos, describiendo actividades de una o más organizaciones. Por ejemplo, las plantillas Productos y Ventas, junto con el archivo de texto Usuarios en nuestro ejemplo de WebShop, es una base de datos. Similarmente, los datos guardados en un banco forman una base de datos, igual como los datos de un hospital, o de una universidad. La manera de manejar una base de datos hoy en día es a través de un sistema de manejo de bases de datos, o un DBMS (abreviación del nombre inglés “Database management system”).
- **Sistema de manejo de bases de datos (DBMS):** Se trata de un software diseñado para ayudar en el manejo de grandes volúmenes de datos, quiere decir, un sistema que permite hacer operaciones sobre una base de datos de manera estructurada, consistente, y eficiente. Un DBMS permite al usuario final encargarse de especificar la estructura de los datos, y los tipos de preguntas que uno va a hacer sobre estos datos, sin preocuparse como los datos son guardados en el disco, como los datos cambian de un día a otro, etc. En este sentido, un DBMS actúa como un servidor central encargado de los datos de una organización, y permite a varios usuarios conectarse y acceder a los datos.

Usuarios del DBMS se dividen en tres grupos principales:

- 1. **Desarrolladores de DBMS:** Los desarrolladores de un DBMS son los ingenieros que implementan el software de DBMS, especificando como los datos son guardados en el disco, como uno accede a los datos de manera eficiente, y como asegurar contra las fallas del sistema.
- 2. **Administradores de DBMS:** Administrador de un DBMS es la persona que especifica qué información se va a guardar en la base de datos, como el usuario final accede a esta información, y que usuario tiene el permiso de acceso.
- 3. **Usuarios finales:** Finalmente, los usuarios finales son las personas que se conectan a un DBMS, y a través las funcionalidades especificadas por su administrador, exploran, analizan, y modifican a los datos.

Ejemplo de un DBMS en el contexto del WebShop:

sería un software que tiene el acceso a todas las plantillas Excel, y a archivos de texto con la información de los usuarios, y permite a cada unidad modificar a sus datos a través este software, sin la necesidad de abrir las plantillas Excel, o el archivo del texto en sí. Este software se preocupa de permitir acceso a los datos solo a las personas autorizadas, y reflejar de manera consistente cualquier cambio a los datos. Si Juan implementó este software en un lenguaje de programación, él es el **desarrollador de la DBMS**. Por otro lado, los gerentes que especifican que datos se guardan en las plantillas Ventas y Productos(i.e. RUT, Nombre...), y el archivo de texto, y quienes deciden si la unidad de ventas puede acceder a una plantilla, son **administradores de la DBMS**. Finalmente, la gente como Manuela y Jorge de la unidad de ventas, quienes hacen cambios al contenido de la plantilla Ventas, son los **usuarios finales**.

Ventajas de un sistema de bases de datos

- **Independencia de los datos:** Los usuarios finales no necesitan saber cómo el DBMS guarda los datos en el disco, sino que provee una vista abstracta de los datos que hay.
- **Eficiencia en el acceso a los datos:** DBMS es un programa especializado para buscar y analizar a los datos. Por esto, un DBMS permite ejecución rápida de estas operaciones, en particular cuando los datos son muy grandes.
- **Integridad de los datos:** DBMS siempre se preocupa que solo hay un usuario con el mismo RUT, que cada dirección tiene un código postal asociado, etc.
- **Administración de acceso:** Un DBMS permite especificar quien tiene acceso a cuáles datos. Además, DBMS permite que dos usuarios usan el mismo dato al mismo tiempo, y asegura que no pasan inconsistencias como en el Ejemplo 2 arriba.
- **Recuperación de fallas:** En el caso que nuestro programa deja de funcionar (por un error de usuario, o por la fuerza mayor), el DBMS permite recuperar el último estado consistente de la base de datos.
- **Reducción del tiempo del desarrollo:** En el caso que uno quiere desarrollar una aplicación que ocupa cierta base de datos para su funcionalidad, usar un DBMS permite omitir todo el trabajo de programar la capacidad de acceder a los datos y modificarlos.
- **Estandarización:** Existe una gran cantidad de DBMS bien establecidos, y usados en la práctica. Por esto, se conoce muy bien su rendimiento en distintos ámbitos, y su utilidad para distintas tareas. Esto también hace nuestros datos fáciles de reutilizar

Debilidades de bases de datos

A pesar de las ventajas, también existen desventajas, tales como:

- **Overhead:** En el caso que trabajamos con datos muy pequeños, integrar a un DBMS en nuestra aplicación puede causar funcionamiento menos eficiente. Igualmente, si vamos a utilizar datos de forma muy específica (por ejemplo, para cambiar letras minúsculas a letras mayúsculas en un texto), no hay mucha gracia en usar un DBMS.
- **Costo:** Muchos DBMS son productos comerciales, y su uso puede incurrir costos no menores. No obstante, hoy en día para la mayoría de los DBMS existe una alternativa open source de muy buena calidad.

Ahora que sabemos que es un sistema de manejo de bases de datos, vamos a introducir el DBMS más usado hoy en día: Bases de datos Relacionales, y sus sistemas de manejo, llamados RDBMS.

Bases de datos relacionales

Las bases de datos relacionales son una tecnología con la que interactuamos todos los días. Por ejemplo, cuando uno hace una compra usando su tarjeta de crédito, es muy probable que la información sobre su transacción se guarde en una base de datos relacional por el proveedor de la tarjeta. Datos de salud, información del censo, estado de cuentas, entre otros, son tipos de datos que se usualmente guardan en una base de datos relacional. En lo que sigue explicaremos qué es una base de datos relacional, y cómo organizar a nuestros datos usando esta tecnología.

Modelo Relacional de los datos

Una base de datos relacional es simplemente un conjunto de tablas (también llamadas relaciones) que representan entidades del mundo real (como por ejemplo persona, direcciones, o productos vendidos por una empresa). Esto nos dice que, en esencia, podemos pensar que se tratan de hojas de Excel que interactúan entre ellas. Por ejemplo, la base de datos relacional para WebShop tenía la tabla Productos:

Código	Nombre	Precio	Stock
123	Sony AC2020	$150,000	2
127	Panasonic KS12	$250,000	0
143	GoPro Hero 4	$199,999	12
229	GoPro Hero 5	$198,999	2
301	Apple Mouse	$1,000,000	5

Cada tabla tiene varias filas y columnas. Las columnas de una tabla se llaman atributos de la relación, y representan características de la entidad. Por ejemplo, los atributos de la relación Productos son: Código, Nombre, Precio, y Stock. Las filas de una tabla/relación se llaman tuplas de la relación. Por ejemplo, lo siguiente es una tupla de la relación Productos:

127	Panasonic KS12	$250,000	0

Formalmente, una relación es simplemente un conjunto de tuplas donde todas tienen los mismos atributos y estos corresponden a los de la relación (el header). La especificación de la estructura de una relación se llama esquema de la relación. El esquema de la relación consta del nombre de la relación, seguido con la lista de los atributos entre paréntesis. Por ejemplo, **Productos**(Código, Nombre, Precio, Stock), es el esquema de la relación Productos. Muchas veces uno también especifica el tipo de datos para cada atributo, por ejemplo, para exigir que el código sea un número, el nombre una palabra, el precio un número, etc.

Esquema Relacional

La especificación de los contenidos de una base de datos relacional se llama **esquema de la base de datos**, y consta de un listado de los esquemas de todas las relaciones que existen en esta base de datos.

Por ejemplo, los datos de WebShop se podrían guardar en una base de datos relacional con el siguiente esquema:

- **Productos**(Código, Nombre, Precio, Stock)
- **Ventas**(RUT, Código, Estatus)
- **Usuarios**(RUT, Nombre, Calle, Comuna, CódigoPostal)

Instancias

Una instancia para un esquema es un conjunto de tuplas para cada relación del esquema. Entonces, una base de datos relacional es siempre una instancia de un esquema. Por ejemplo, una instancia de la base de datos de WebShop con el esquema especificado arriba sería:

Productos

Código	Nombre	Precio	Stock
123	Sony AC2020	$150,000	2
127	Panasonic KS12	$250,000	0
143	GoPro Hero 4	$199,999	12

Ventas

RUT	Código	Status
18.268.BBB – 7	123	Pagado

Usuarios

RUT	Nombre	Calle	Comuna	Código Postal
15.378.AAA - K	Gonzalo Morales	Claudio Montt 101	Providencia	750000
18.268.BBB - 7	Gonzalo Arenas	Av. Diego Varas 123	Providencia	750000

Ventajas de un RDBMS EN WebShop

La gran ventaja de utilizar una base de datos relacional para WebShop es que nos permite evitar los problemas mencionados antes. Primero, el problema de acceso concurrente a los datos es manejado de manera automática por el DBMS. Esto quiere decir que cuando Jorge ingresa el pago de Gonzalo al sistema, Andrea ya no puede modificar este dato. Similarmente, si dos personas hacen el pedido para un artículo de cual queda solo una unidad, uno de estos pedidos será procesado primero en el sistema y el otro pedido no se podrá realizar. Segundo, la integración de los datos es fácil cuando las empresas involucradas utilizan bases de datos relacionales. Por ejemplo, uno simplemente puede manejar dos bases de datos (una para WebShop, y la otra para el VegShop) con el mismo DBMS. Finalmente, el DBMS nos permite buscar y modificar a nuestros datos de manera eficiente cuando los datos crecen mucho y nuestras relaciones cuentan con miles de tuplas. Quizás lo más importante es que los usuarios de nuestra base de datos no necesitan preocuparse como los datos son guardados en el disco, como navegar hacia persona con cierto RUT, como se reflejan los cambios en la base de datos, ni qué pasa si dos personas intentan acceder al mismo dato al mismo tiempo. El encargado de todas estas tareas será nuestro sistema de manejo de bases de datos.

Diseño de bases de datos relacionales

La pregunta más importante cuando uno usa bases de datos relacionales es: ¿Cómo diseñar un buen esquema para mi base de datos relacional? Aunque esto es un tema muy complejo, y la mayoría de las bases de datos tienen que cambiar su esquema durante su ciclo de vida útil para mejorar su rendimiento, aquí vamos a explicar unas reglas básicas para el diseño de un buen esquema.

Primero…

uno tiene que decidir si es necesario usar un DBMS para su aplicación. Como hemos visto antes, un DBMS tiene muchas ventajas comparado con manejar los datos “a mano“, en particular cuando uno tiene muchos usuarios, y maneja un sistema muy complejo. Pero por otro lado, si Juan tiene una empresa pequeña y lo único que tiene que guardar son los datos de identificación de sus empleados, y el sueldo que les paga, le sirve mucho más manejar esta información en una plantilla Excel que tenga el RUT, Nombre y Sueldo. Usar un motor de bases de datos en este contexto sería un “overkill“.

Luego…

Si ya hemos decidido usar una base de datos relacional, el siguiente paso es diseñarla. Una de las herramientas más importantes para esta tarea son las restricciones de integridad. Aunque existen muchas restricciones de integridad, la más común y usada son las llaves. La llave en una relación es el atributo que nos permite identificar de manera única una tupla de la relación. Intuitivamente, la llave en una relación es lo mismo como el RUT de una persona: es el valor que nos permite identificar al dato. Por ejemplo, si miramos a la relación Usuarios en nuestra base de datos para WebShop, uno quiere que el RUT de la persona sea configurado como llave para esa relación. Esto quiere decir que no pueden existir dos filas con el mismo RUT en nuestra relación, porque el RUT determina a la persona y fila de manera única.

Si el RUT es la llave de la relación Usuarios, lo siguiente es una instancia válida de la relación:

Usuarios

RUT	Nombre	Calle	Comuna	Código Postal
15.378.AAA - K	Gonzalo Morales	Claudio Montt 101	**Providencia**	**750000**
18.268.BBB - 7	Gonzalo Arenas	Av. Diego Varas 123	**Providencia**	**750000**

Pero, lo siguiente no lo es:

Usuarios

RUT	Nombre	Calle	Comuna	Código Postal
15.378.AAA - K	Gonzalo Morales	Claudio Montt 101	**Providencia**	**750000**
15.378.AAA - K	Gonzalo Arenas	Av. Diego Varas 123	**Providencia**	**750000**

Llaves compuestas

En estricto rigor, la llave no es necesariamente solo un atributo, sino que puede constar de más de un atributo. Llaves de este tipo de llaman llaves compuestas. Una llave compuesta es simplemente un conjunto de dos o más atributos en nuestra relación que actúan como la llave. Por ejemplo, si WebShop tuviese una unidad de recursos humanos, para guardar información sobre las liquidaciones mensuales de los sueldos, debería manejar una tabla de este estilo:

Sueldos

RUT	Mes	Año	Monto
15.378.AAA - K	Junio	2019	2.500.000
15.378.AAA - K	Agosto	2019	2.450.000

Si designamos solo el RUT como la llave de la relación Sueldos, esto permitiría pagar a un empleado solo una vez, porque la llave no permitiría insertar más tuplas con el mismo RUT. Para evitar esta situación, podemos designar el conjunto (RUT, Mes, Año) como la llave compuesta de la relación Sueldos. De esta forma para cada mes del año se puede registrar un único pago de sueldo a un empleado.

Como nuestro último ejemplo, vamos a considera la relación Ventas arriba. La pregunta sería cuál es una buena llave para esta relación. Aquí tenemos varias opciones, pero todas con sus problemas:

1. **RUT es la llave:** Usando esta solución, un usuario podría realizar solo una compra, porque el sistema no nos permite ingresar otra tupla con el mismo RUT.

2. **Código es la llave:** Tiene el mismo problema como en el ítem uno, solo que dos usuarios no pueden comprar artículos con el mismo código.

3. El **par (RUT,Código) es la llave:** En este caso, un usuario puede comprar el ítem con cierto código solo una vez.

4. **La tupla entera (RUT,Código,Status) es la llave:** Esta solución tiene el mismo problema que la opción anterior.

Sugieran entre estas 4 opciones cuál es la mejor…

Este ejemplo nos muestra que la relación Ventas es mal diseñada, porque no permite que se defina una llave útil para la relación. En realidad, el diseño de la relación Ventas estuvo mal desde el inicio, ya que se debería permitir distinguir cada compra como una acción separada y única. La solución que se usa en la práctica para distinguir distintas transacciones monetarias es generar un identificador único para cada transacción. Esto quiere decir que nuestra relación Ventas se debería ver de siguiente manera:

Ventas

TransID	RUT	Código	Status
1	18.268.BBB – 7	123	Pagado
2	18.268.BBB – 7	123	Pendiente
28	18.268.BBB – 7	143	Pendiente

Finalmente…

Para resaltar los atributos que forman la llave de una relación, se subraya su nombre en el esquema. Por ejemplo, el esquema correcto para nuestra base de datos de WebShop, que permite manejo correcto de las transacciones es lo siguiente:

- **Productos**(Código, Nombre, Precio, Stock)
- **Ventas** (TransId, RUT, Código,Estatus)
- **Usuarios**(RUT, Nombre, Calle, Comuna, CódigoPostal).

Conclusión de la clase…

En esta clase discutimos la importancia de manejar nuestros datos de manera estructurada y ordenada. Para esto, mostramos el ejemplo de una empresa que con su crecimiento enfrentaba problemas más complejos con el manejo de sus datos. También, introdujimos la tecnología más usada para enfrentar estos problemas: las bases de datos relacionales. Ahora deberíamos entender bien las principales ventajas de manejar nuestros datos a través de un motor para el manejo de bases de datos, como se describe una base de datos relacional, y algunas reglas de buen diseño de bases de datos relacionales

Bibliografía

- Ramakrishnan, R., Gehrke, J., Database Management Systems, 3rd edition, McGraw-Hill, 2002.

- Soto, A., Bases de Datos, https://github.com/alanezz/Syllabus-2019-1, 2019