1.- ¿Qué es una base de datos?

En esencia una base de datos no es más que una colección de información que existe durante un largo período de tiempo, a menudo muchos años.

2.- ¿Qué es un SGBD?

Un DBMS se caracteriza por la capacidad de soportar un acceso eficiente a grandes cantidades de datos, que persiste en el tiempo. También se caracteriza por el apoyo de los lenguajes de consulta de gran alcance y transacciones duraderos que pueden ejecutar al mismo tiempo de una manera que parece atómica e independiente de otra transacción.

2.1.- SGBD tempranos

El primer sistema de gestión de base de datos comercial apareció en 1960. Estos sistemas evolucionaron a partir de los sistemas de archivos.

Algunas de las aplicaciones de estos sistemas era: los sistemas de avión, sistemas bancarios, registros corporativos.

2.2.- SGBD

Ted Codd en 1970 cambió de manera significativa los sistemas de bases de datos. Codd propuso que los sistemas de bases de datos deben presentar al usuario con una vista de datos organizados como tablas llamadas relaciones.

SQL (Structured Query Language) es el lenguaje de consultas más importante basado en el modelo relacional.

Para 1990, los sistemas de bases de datos relacionales eran la norma. Sin embargo, el campo de base de datos sigue evolucionando, y los nuevos temas y enfoques para la gestión de superficie de datos con regularidad.

2.3.- Esquema de los estudios de los SGBD

DISEÑO DE BD:

1. ¿Cómo se desarrolla una base de datos útil?
2. ¿Qué tipo de información entrar en la base de datos?
3. ¿Cómo se estructura la información
4. ¿Qué suposiciones se hacen acerca de los tipos o valores de elementos de datos?
5. ¿Cómo se conectan los elementos de datos?

BASE DE DATOS DE PROGRAMACIÓN:

1. ¿Cómo se expresan las consultas y otras operaciones en la base de datos?
2. ¿Cómo un uso otras capacidades de un DBMS, como transacciones o limitaciones, en una aplicación?
3. ¿Cómo se la programación de base de datos combina con programación convencional?

SISTEMA DE BASE DE DATOS DE APLICACIÓN:

1. ¿Cómo se puede construir un DBMS, incluyendo aspectos tales como el procesamiento de consultas, el procesamiento de transacciones y la organización de almacenamiento para el acceso eficiente?

1.- Introducción a las bases de datos

Las bases de datos actuales son esenciales para todas las empresas. Se utilizan para mantener registros internos, para presentar los datos a los clientes y clientes en la World Wide Web, y para apoyar a muchos otros procesos comerciales.

Las bases de datos están igualmente encontrar en el centro de muchas investigaciones científicas. Representan los datos recogidos por los astrónomos, por investigadores del genoma humano, y por los bioquímicos explorar las propiedades medicinales de las proteínas, junto con muchos otros científicos.

2.- Bases de datos y Sistemas Gestores de Bases de Datos (SGBD)

2.1.- Los datos y las bases de datos

Hay tres ámbitos o mundos de datos:

- **El mundo real:** son los objetos de la realidad, y cuyo será gestionar.
- **El mundo conceptual:** Establecer los conocimientos obtenidos en el mundo real. Estos conjuntos depende del espectador.
- **El mundo representaciones:** es el conjunto de representaciones de computadoras del mundo. Ellos están obligados a trabajar con datos.

2.2.- Los datos y su representación

Los datos son la representación informática de la información disponible. Hace referencia a interesante para nosotros los objetos del mundo real. REPRESENTACIÓN MUNDIAL está configurado por los datos informatizados trabajadas.

Dos etapas son necesarias con el fin de convertir el mundo real a los datos de la computadora; Ellos son:

1. **Diseño Lógico:** Funciona con el modelo abstracto de datos. Ese modelo se obtiene al final de la etapa de modelo conceptual. El objetivo de esta etapa es traducir el mundo real al modelo de datos utilizado por DBMS.
2. **Diseño físico:** Es la etapa en la que se aumenta la eficiencia de la operación.

2.3.- Entidad, atributo y valor

La información se define por tres elementos:

1. **Entidad:** La palabra ENTIDAD es un objeto de la realidad conceptualizada y estamos interesados en algunas características de la misma. Hay dos entidades: Tipo de Entidad es una palabra genérica utilizada en el lenguaje real y solicitud de la entidad es un objeto específico. Por ejemplo: coche es un tipo de entidad. Su coche con ID es 1234 BCD es una Solicitud de Entidad.
2. **Atributo:** propiedades de la entidad que estén interesados. Ejemplo: el coche entidad 1234 BCD es tipo, color, vidrio, hp, cc.
3. **Valor:** son los valores de los atributos. Ejemplo: el coche entidad 1234 BCD es deportivo, negro, oscuro, 147hp, 3.5 cc.

Acerca ENTIDAD hay dos tipos:

- **Tipo de entidad:** es una entidad genérica o, más correctamente, es una abstracción de un verdadero cosas establecidas. F.i .: un coche
- **Entidad Instancia:** es un objeto de conceptualización del mundo real. F.i .: la placa del carnet de conducir.

3.- Atributos, tipos de datos y dominios

Todo el valor establecido que un atributo puede tomar se llama dominio.

Un tipo de datos definen un valor ajustado con algunas características comunes, estas características hacen compatibilidades, a continuación, sobre las operaciones que podrían definirse.

Ejemplo de tipo de datos

Deje conjunto entero sea como un tipo de datos. Operación como más, menos, los tiempos se puede definir, pero la división exacta no se puede definir en este conjunto.

Ejemplo de tipo de dominio

Se tiene en cuenta la licencia de conducir español (DL). Cuando pierdes todos los puntos, el DL se revoca y se debe pasar el examen en otro momento. El número de puntos es de 0 a 15, esto es el dominio en este caso.

4.- El valor NULL

El valor nulo expresión indican que cualquier valor está asociado a ese atributo de una entidad. Por ejemplo, una persona que no tienen el DL, no tiene puntos, ni 0.

5.- Claves y atributos ID

Un atributo identificador (ID) permite identificar inequívocamente una entidad del resto. Valor Es único. Por ejemplo, el número de tarjeta de identificación es única para todo el mundo.

El conjunto de atributos de identidad se llama llaves. Por ejemplo: El código postal es un elemento clave en las compras de los grandes almacenes.

6.- Representación de tablas y su implementación

La información es el resultado de analizar y de hacer que los conceptos del mundo real. Hay representaciones que no son eficientes para ver la información, y es necesario, utilice una buena representación. El más utilizado es la representación tabular.

En la representación tabular cada fila representa una entidad ejemplo, cada columna representa un atributo y cada célula contiene el valor pertinente.

La mayoría de estas características, sin embargo, dependen, al menos en parte, de un buen diseño de base de datos. Si no elaborar un buen diseño, se perderá en alguna de los beneficios de estas características.

A continuación se describen algunas de las características que un buen sistema de base de datos debe proporcionar y explicar en qué medida dependen de un buen diseño de base de datos.

1.- CRUD

CRUD representa las cuatro operaciones de bases de datos fundamentales que cualquier base de datos debe proporcionar:
Crear, Leer, Actualizar y Eliminar.

CRUD es más una característica de las bases de datos en general de lo que es una característica de un buen diseño de base de datos, pero un buen diseño de base de datos proporciona CRUD eficiente.

Cuando se crea un nuevo registro (la C en CRUD), la base de datos debe validar la nueva entrada del Estado. De manera similar al actualizar un registro (la U en CRUD), la base de datos debe validar la modificación Entrada Estado.

Cuando se elimina una entrada en la tabla Unidos (el D en CRUD), la base de datos debe verificar que no Registros de los participantes utilizan ese estado. Por último, cuando usted lee los datos (la R en CRUD), el diseño de bases de datos determina si usted encuentra los datos que desee en cuestión de segundos, horas o no en absoluto.

2.- Recuperación

Recuperación es otra palabra para " leer ", la R en CRUD. La base de datos debe permitir a encontrar cada pieza de datos. No tiene sentido poner algo en la base de datos si no hay manera de recuperarlo después. (Eso sería un " agujero negro de datos, " no es una base de datos.)

La base de datos debe permitir a estructurar los datos para que pueda encontrar piezas particulares de datos en uno o más específicas maneras.

Idealmente, la base de datos también le permitirá estructurar los datos por lo que es relativamente rápida y fácil de obtener los datos de una manera particular.

Por el contrario, es probable que no es necesario buscar a los clientes por el segundo nombre con demasiada frecuencia.

3.- Consistencia

Otro aspecto de la R en CRUD es consistencia. La base de datos debería proporcionar resultados consistentes. Si se realiza la misma búsqueda dos veces seguidas, usted debe obtener los mismos resultados. Otro usuario que realiza la misma búsqueda también debe obtener los mismos resultados.

A well-built database product can ensure that the exact same query returns the same result but design also plays an important role. If the database is poorly designed, you may be able to store conflicting data in different parts of the database.

4.- Validación

Validez está estrechamente relacionado con la idea de consistencia. Coherencia significa diferentes partes de la base de datos no tienen puntos de vista contradictorios de la misma información. Validez significa que los datos se valida cuando sea posible frente a otras piezas de datos en la base de datos. En términos CRUD, los datos pueden ser validados cuando se crea un registro, actualizan o eliminan.

Al igual que los contenedores de datos físicos, una base de datos informatizada puede contener datos incompletos, inexactos o contradictorios. Nunca se puede proteger una base de datos de los usuarios que no pueden deletrear o que acaba de entrar en llano la información equivocada, pero un buen diseño de base de datos puede ayudar a prevenir algunos tipos de errores que una base de datos física no puede prevenir.

La base de datos también puede verificar que un valor introducido por el usuario está presente en otra parte de la base de datos. La base de datos también puede comprobar algunos tipos de condiciones en los datos.

5.- Fácil conexión de error

Incluso una base de datos perfectamente diseñado no puede asegurar la validez perfecto. ¿Cómo puede la base de datos de saber que el nombre de un cliente se supone que debe ser deletreado Pheidaux no Fido como tecleado por el usuario?

La corrección de un solo error en un cuaderno es bastante fácil. Sólo hay que cruzar el valor incorrecto y escribir en la nueva.

Corrección de errores sistemáticos en un portátil es mucho más difícil. Supongamos que usted contrata a un pasante de verano para ir de puerta en puerta vendiendo productos para el hogar y escribe una gran cantidad de órdenes de " cinta pato " sin darse cuenta de que el producto real es " la cinta aislante. " La fijación de todos los errores podría ser tedioso y consume mucho tiempo.

En una base de datos informatizada, este tipo de corrección es trivial. Un comando de base de datos simple puede actualizar todas las apariciones del nombre del producto " cinta pato " a lo largo de todo el sistema.

6.- Velocidad

Un aspecto importante de todos los componentes CRUD es la velocidad. Una base de datos bien diseñada puede crear, leer, actualizar y eliminar registros rápidamente.

No se puede negar que una base de datos informatizada es mucho más rápido que un portátil o un archivador. En lugar de procesar a decenas de registros por hora, una base de datos informatizada puede procesar decenas o cientos por segundo.

Un buen diseño juega un papel crítico en la eficiencia de la base de datos. Una base de datos mal organizados todavía puede ser más rápido que el documento equivalente, pero será mucho más lento que una base de datos bien diseñada.

7.- Transacciones atómicas

Recordemos que una transacción atómica es un posiblemente compleja serie de acciones que se considera como una sola operación por aquellos que no participan directamente en la ejecución de la transacción. Si transfiere $ 100 de la cuenta de Alice a la cuenta de Bob, nadie más puede ver la base de datos mientras se encuentra en un estado intermedio donde el dinero se ha eliminado de la cuenta de Alice y aún no añadido a Bob.

La transacción ya sea pasa por completo o ninguna de sus piezas suceda - que no puede suceder hasta la mitad.

Transacciones atómicas son importantes para mantener la consistencia y validez, por lo que son importantes para la partes U de CRUD R y. Contenedores de datos físicos, tales como cuadernos soportan transacciones atómicas, porque normalmente sólo una persona a la vez puede utilizarlos.

Estas bases de datos también rollback automáticamente cualquier transacción que está abierto, si la base de datos se detiene de forma inesperada.

8.- ACID

Esta sección proporciona más detalles sobre las operaciones descritas en el apartado anterior en lugar de hablar de una nueva función de contenedores de datos físicos y bases de datos informatizadas.

ACID es un acrónimo que describe cuatro características que un sistema de transacción de efectivo debe proporcionar. ACID significa atomicidad, consistencia, aislamiento y durabilidad. Atomicidad significa transacciones son atómicas. Las operaciones en una transacción o bien todos suceden o ninguno de ellos ocurren.

La consistencia significa la transacción se asegura de que la base de datos está en un estado coherente antes y después de la transacción. En otras palabras, si las operaciones dentro de la transacción violarían las reglas de la base de datos, la transacción se deshace. Aislamiento significa la transacción aísla los detalles de la transacción a todos, salvo la persona que realiza la transacción. Durabilidad significa que una vez que se confirma una transacción, no va a desaparecer después. Si la energía falla, cuando la base de datos se reinicia, los efectos de esta transacción aún estarán allí.

9.- Persistencia copias de seguridad

Los datos deben ser persistente. No debe cambiar o desaparecer por sí mismo. Si usted no puede confiar en la base de datos para mantener los datos seguros, la base de datos es más o menos valor. Productos de base de datos hacen todo lo posible para mantener los datos seguros y en funcionamiento normal no es necesario hacer mucho para obtener el beneficio de la persistencia de datos.

10.- Bajo costo y extensibilidad

Lo ideal sería que la base de datos debe ser fácil de obtener e instalar, de bajo costo y fácilmente extensible. Si usted descubre que usted necesita para procesar muchos más datos por día de lo que había esperado, usted debería ser capaz de aumentar de alguna manera la capacidad de la base de datos. Aunque algunos productos de base de datos son bastante caros, la mayoría de ellos tienen rutas de actualización razonables para que pueda comprar la licencia menos costosa que se encargará de sus necesidades, por lo menos al principio.

11.- Fácil de usar

Cuadernos y archivadores tienen interfaces de usuario simples por lo que casi cualquier persona puede utilizar de manera efectiva. Interfaz de usuario de una aplicación informática determina cómo utilizable es por los usuarios promedio. Diseño de la interfaz de usuario no es parte de la base de datos de diseño, por lo que puede preguntarse por qué la facilidad de uso se menciona aquí.

12.- Portabilidad

Una base de datos informatizada permite una portabilidad que es aún más poderoso que la portabilidad de un portátil. Se le permite acceder a los datos desde cualquier lugar que tenga acceso a la Web sin tener que mover la base de datos física. Se puede acceder a la base de datos desde casi cualquier lugar, mientras que los datos en sí se mantiene de forma segura en casa, lejos de los peligros de los carteristas, que se dejó caer en un charco, y conseguir olvidado en el autobús.

13.- Seguridad

Un cuaderno es relativamente fácil perder o robar, pero una base de datos altamente portátil puede ser aún más fácil hacer concesiones. Si puede acceder a su base de datos de todo el mundo, entonces también lo puede bandidos cibernéticos y otros bueno para nada pozos.

El bloqueo de la base de datos es sobre todo un problema de seguridad que se debe abordar mediante el uso de sus redes y bases de datos herramientas de seguridad. Sin embargo, hay algunas técnicas de diseño que se pueden utilizar para hacer asegurar la base de datos más fácil.
Si separa los datos en categorías que los diferentes tipos de usuarios necesitan manipular, puede otorgar diferentes niveles de permiso para los diferentes tipos de usuarios.
Otro aspecto novedoso para la seguridad de base de datos es el hecho de que los usuarios pueden acceder a la base de datos de forma remota sin llegar a la celebración de una copia de la base de datos a nivel local.

Esto es más un problema arquitectura de la aplicación de un problema de diseño de base de datos (no almacenar los datos localmente en las computadoras portátiles), pero utilizando un diseño de base de datos que restringe el acceso de los usuarios a lo que realmente necesitan saber puede ayudar.

14.- Compartir

No es fácil de compartir un cuaderno o sobre lleno de tarjetas de visita, entre un montón de gente. No hay dos personas realmente pueden usar un cuaderno al mismo tiempo, y hay algo de sobrecarga en el envío de la portátil de ida y vuelta entre los usuarios.

Tomarse el tiempo para caminar por la habitación de una docena de veces al día sería molesto; expresar correo un cuaderno todo el país todos los días sería simplemente una tontería.

Redes modernas pueden permitir que cientos o incluso miles de usuarios acceden a la misma base de datos al mismo tiempo desde lugares repartidos por todo el globo. Aunque esto es en gran parte un ejercicio de creación de redes y las herramientas que proporciona un producto de base de datos particular, algunos problemas de diseño entran en juego.

15.- Capacidad para realizar cálculos complejos

En comparación con el cerebro humano, las computadoras son idiotas. Se necesita hardware serio potente y algoritmos terriblemente sofisticados para llevar a cabo las tareas que usted toma por sentado, como el reconocimiento de rostros, reconocimiento de voz independiente del hablante, y el reconocimiento de escritura.

El cerebro humano es también auto-programación, por lo que puede aprender nuevas tareas de manera flexible y relativamente rápido.

Cuando se trata de equilibrar chequeras, en busca de cuentas con saldos de menos de cero, y la realización de una serie de otras tareas de procesamiento de números, el equipo es mucho más rápido.

El ordenador es naturalmente más rápido en este tipo de cálculos, pero incluso su velocidad de vértigo no le ayudará si su base de datos está mal diseñado. Un buen diseño puede hacer la diferencia entre encontrar los datos que necesita en cuestión de segundos en lugar de horas, días o no en absoluto.

1.- Bases de datos relacionales

Sin entrar en demasiados detalles, una base de datos relacional contiene tablas que contienen filas y columnas. Cada fila contiene datos relacionados sobre una entidad en particular (persona, vehículo, bocadillo, o lo que sea). Cada columna representa un dato acerca de esa entidad (nombre, dirección, número de encurtidos, etc.).

A veces una pieza de información, naturalmente, tiene más de un valor. Por ejemplo, un solo cliente puede colocar un montón de órdenes. Para que sea más fácil agregar varios valores, esos valores se almacenan en una tabla independiente vinculado al primero por algún valor que los registros correspondientes comparten.

Por ejemplo, supongamos que usted construye una base de datos relacional para el seguimiento de sus corredores de luge calle favoritos. El Racers tabla almacena información sobre los corredores individuales. Cada fila corresponde a un piloto en particular. Las columnas representan la información básica para un corredor como nombre, edad, altura, peso, y así sucesivamente. Una columna muy importante almacena el número de identificación de cada corredor.

Con el tiempo, cada corredor tendrá una gran cantidad de resultados de la carrera (aunque es probable que haya un montón de puntos en blanco para chuckers pacas - ver www.skateluge.com/lugetalk.htm). Puede almacenar resultados de la carrera en una mesa RaceResults separada. Cada fila registra la clasificación final para un solo piloto en una sola carrera. Las columnas registrar el número del corredor de identificación, nombre y fecha de la carrera, posición final del corredor, y los puntos de esa posición es digno de la clasificación general.

Para encontrar todas las posiciones finales y puntos para un piloto en particular, se mira la fila del corredor de la tabla Racers, encontrar el número de identificación del corredor y, a continuación, encontrar todas las filas de la tabla RaceResults que tienen este ID corredor.

Racers Table

RacerName	Nationality	RacerId
Michael Serek	Austria	1
Chris McBride	United States	2
Sebastien Tournissac	France	3

RaceResults Table

RaceName	Division	Dates	RacerId	FinishingPosition	Points
Go Fast Speed Days	Pro Classic Luge Mass	9/1/2007-9/2/2007	1	1	450.0024
Rock and Roll	Pro Classic Luge Mass	7/27/2007-7/28/2007	1	1	450.0024
Almabtrieb World Championships	Pro Classic Luge Mass	7/11/2007-7/14/2007	1	6	403.3633
Almabtrieb World Championships	Pro Classic Luge Mass	7/11/2007-7/14/2007	2	24	321.1366
Almabtrieb World Championships	Pro Classic Luge Mass	7/11/2007-7/14/2007	3	2	432.6154
Go Fast Speed Days	Pro Classic Luge Mass	9/1/2007-9/2/2007	2	2	432.6154
Go Fast Speed Days	Pro Classic Luge Mass	9/1/2007-9/2/2007	3	3	424.3687
Top Challenge	Pro Street Luge Mass	8/25/2007-8/26/2007	2	13	0

Bases de datos relacionales han existido por mucho tiempo. (Edgar Codd comenzó sentar las bases en 1970.) Ellos son el tipo más común de base de datos de hoy y han sido durante años, por lo que una gran cantidad de empresas muy poderosas han gastado una enorme cantidad de tiempo que la construcción de ellos. Todo esto significa que bases de datos relacionales se han estudiado a fondo y han evolucionado con el tiempo hasta el punto en que son muy útiles y eficaces.

Bases de datos relacionales ofrecen una serie de características que hacen que trabajar con bases de datos tales como la base de datos luge de la calle más fácil. Algunas de las características que ofrecen son:

- **Los tipos de datos:** Cada columna tiene un tipo particular de datos (texto, numérico, fecha, y así sucesivamente) y la base de datos no permitirá valores de otros tipos en una columna.
- **Restricciones básicas**: La base de datos puede hacer cumplir las restricciones como la exigencia de que la velocidad máxima del corredor luge estar entre 50 y 250 mph (nadie con una velocidad máxima de menos de 50 vale la grabación) o puede requerir ciertos campos.
- **La integridad referencial:** La base de datos puede evitar que la adición de un registro RaceResults para un corredor que no existe en la tabla Racers. Del mismo modo, la base de datos puede impedir la modificación de Identificación de un corredor si eso dejaría filas en la tabla RaceResults con identificaciones corredor no válidos, y podría evitar que la modificación del corredor Identificación de una fila RaceResults a un valor no válido.
- **Eliminaciones en cascada y actualizaciones:** Si elimina un corredor de la tabla de los corredores, la base de datos puede eliminar automáticamente todos RaceResults registros de ese corredor. Del mismo modo, si usted cambia el número de identificación de un corredor, la base de datos puede actualizar los números de identificación en RaceResults registros de ese corredor.
- **Combinaciones:** La base de datos puede reunir rápidamente los registros relacionados de diferentes tablas. Por ejemplo, se puede enumerar fácilmente cada corredor con sus posiciones finales correspondientes ordenados alfabéticamente y por fecha raza.
- **Consultas complejas**: las bases de datos relacionales soportan todo tipo de consultas y de agregación interesantes funciones como SUM, AVG, MIN, COUNT, STDEV y GROUP BY.

Bases de datos relacionales funcionan bien si:

- Es necesario realizar consultas complicadas y se une entre las diferentes mesas.
- Es necesario para realizar validaciones de datos, tales como la verificación de la existencia de filas relacionadas en otras mesas.
- Es necesario para permitir cualquier número de valores para una determinada pieza de datos (por ejemplo, la raza acabado posiciones).
- ¿Quieres ser capaz de construir con flexibilidad las nuevas consultas que usted no piensa cuando inició el diseño del proyecto.

Bases de datos relacionales no funcionan bien si:

- Es necesario utilizar una topología de datos especial para llevar a cabo la función principal de la aplicación. Por ejemplo, usted puede batir una jerarquía o red con un ladrillo hasta que encaje en una base de datos relacional pero puede obtener un mejor rendimiento utilizando un tipo más especializado de base de datos.

A menos que tenga necesidades especiales, bases de datos relacionales suelen ser una excelente opción.

2.- XML

XML (eXtensible Markup Language) es un lenguaje para almacenar datos jerárquicos. XML en sí no ofrece herramientas para la construcción, búsqueda, actualización, validación, o de otra manera manipular datos y cualquiera que le diga lo contrario está tratando de vender algo.

Conceptos básicos de XML

Un archivo XML es un archivo de texto relativamente simple que utiliza fichas especiales para definir una estructura para los datos que contiene. La gente suele comparar XML para el

lenguaje Web HTML (HyperText Markup Language), ya que ambas señales de uso rodeadas por corchetes puntiagudas, pero los dos idiomas tienen varias diferencias grandes.

Estructuras XML

En la práctica que normalmente veo archivos XML utilizan con mayor frecuencia en una de tres maneras.

En primer lugar, los archivos XML son jerárquicos por lo que es natural para los utilizan para almacenar datos jerárquicos. Es fácil de mapear datos puramente jerárquicas como un árbol genealógico simple o organigrama en un archivo XML. En segundo lugar, los archivos XML se utilizan a menudo para mantener los datos de mesas similares. La estructura básica sigue de cerca la estructura de una base de datos relacional. El elemento raíz contiene varios elementos de la tabla. Cada uno de esos elementos bodegas "registros" que tienen "campos". Por ejemplo, el siguiente documento XML contiene datos sobre los clientes de un sencillo de la empresa y sus órdenes:

```
<AllData>
	<Customers>
		<Customer ID="1">
			<FirstName>Alfred</FirstName>
			<LastName>Gusenbauer</LastName>
		</Customer>
		<Customer ID="2">
			<FirstName>David</FirstName>
			<LastName>Thompson</LastName>
		</Customer>
		<Customer ID="3">
			<FirstName>Alberto</FirstName>
			<LastName>Selva</LastName>
		</Customer>
	</Customers>
	<Products>
		<Product ID="273645" Description="Toothbrush" Price="$1.95" />
		<Product ID="78463" Description="Pencil" Price="$0.15" />
		<Product ID="48937" Description="Notepad" Price="$0.75" />
	</Products>
	<CustomerOrders>
		<CustomerOrder Date="12/27/2008" CustomerId="2">
			<Item ID="1" ProductId="78463" Quantity="12" />
			<Item ID="2" ProductId="48937" Quantity="2" />
		</CustomerOrder>
	</CustomerOrders>
</AllData>
```

El archivo comienza con un elemento raíz AllData. Ese elemento contiene tres elementos más que definen las estructuras de mesa como la celebración de cliente, producto y la información de pedidos de clientes. Cada uno de estos " tablas " define " registros ". Por ejemplo, el elemento de clientes incluye a cliente " registros " que mantienen valores nombre y apellidos. Este documento XML utiliza los números de identificación de vincular los registros en diferentes " tablas " juntos. En este ejemplo, el elemento CustomerOrder sola representa un pedido realizado por el cliente 2 (David Thompson) que ordenó 12 artículos con el ID 78463 (lápices) y 2 elementos con ID 48937 (libretas). La tercera estructura de archivos XML que he visto con regularidad es una simple lista de valores. El siguiente documento XML utiliza esta estructura para sostener los valores de configuración de una aplicación:

```
<Settings>
	<NormalColor>Black</NormalColor>
	<WarningColor>Green</WarningColor>
	<ErrorColor>Yellow</ErrorColor>
	<PanicSound>panic.wav</PanicSound>
	<BugEmail>bugs@panic.com</BugEmail>
</Settings>
```

Este tipo de archivo XML da un poco más estructura que un archivo de texto plano que se utiliza para mantener la configuración y permite que un programa de uso de herramientas XML para cargar fácilmente y leer los valores de ajuste.

Archivos XML funcionan bien si:

1. Los datos son naturalmente jerárquica.
2. Herramientas de XML disponibles proporcionan las características que necesita.
3. Usted quiere que los tipos de validación que los archivos de esquema pueden proporcionar.
4. Usted desea importar y exportar los datos en los productos que entienden XML.

Archivos XML no funcionan bien si:

1. Utilice los datos no jerárquicas, como las redes (que se describen en la siguiente sección).
2. Usted necesita la validación de datos más complejos que los archivos de esquema pueden proporcionar.
3. Es necesario realizar relacional en lugar de consultas jerárquicas.
4. La base de datos es muy grande por lo que volver a escribir todo el archivo para actualizar un pequeño bit de datos en el medio es engorroso.
5. Es necesario para permitir que varios usuarios actualizar con frecuencia la base de datos sin interferir entre sí.

Usted puede encontrar gran cantidad de tutoriales gratuitos que cubren XML y sus tecnologías relacionadas, como XSL, XSLT, XPath, XQuery, y otros en el sitio Web W3 Schools (www.w3schools.com).

3.- Objeto-relacional

Una base de datos objeto-relacional (ORD) o el sistema de gestión de bases de datos objeto-relacional (ORDBMS) es una base de datos relacional que ofrece características adicionales para la integración de los tipos de objetos en los datos. Al igual que una base de datos relacional, puede realizar consultas complejas con relativa rapidez. Al igual que una base de datos de objetos, que utiliza una sintaxis especial para simplificar la creación de los objetos.

Con el tiempo, muchas de las características originalmente diseñados para su uso por las bases de datos objeto-relacionales se han añadido a las bases de datos relacionales.

Objeto-relacional de bases de datos y el objeto-relacional asignaciones funcionan bien si:

1. El entorno de programación y favores arquitectura utilizando objetos.
2. Es necesario realizar consultas de tipo relacional complicados.
3. Es necesario para realizar validaciones de datos de tipo relacional.
4. Su programa necesita para interactuar con herramientas externas, donde el almacenamiento de los datos en un formato relacional común es una ventaja.
5. Usted tiene programadores independientes y desarrolladores de bases de datos por lo que el mantenimiento de una estricta separación puede hacer que el proyecto sea más manejable.

Objeto-relacional de bases de datos y el objeto-relacional asignaciones no funcionan bien si:

- No está utilizando un lenguaje orientado a objetos (por ejemplo, si una base de datos Microsoft Access puede hacer todo lo que necesita sin necesidad de programación).