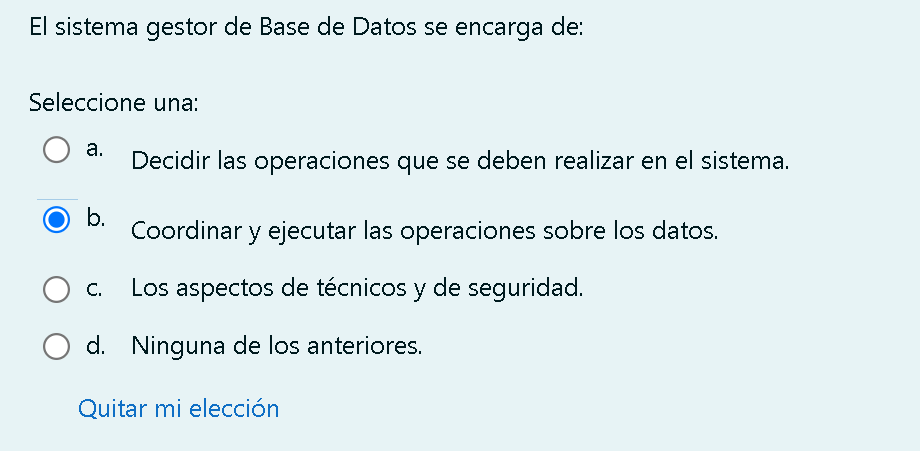
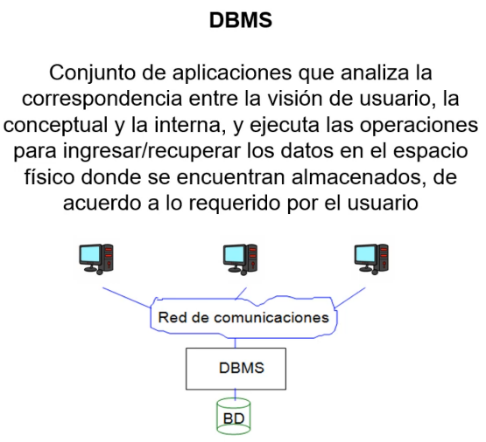


Un sistema de base de datos está integrado por usuarios, que son los que decidirán las operaciones que debe realizar el sistema gestor, como ingreso de datos, actualizaciones, eliminación, consultas, etcétera; aplicaciones de software que son sistemas que usan los datos almacenados en la base de datos. Generalmente están orientados al negocio, por ejemplo, ventas, compras, pagos; hardware que son los servidores donde estará instalado el sistema gestor y el espacio donde se almacenarán los datos; sistema gestor de la base de datos, que como ya dijimos, es la aplicación que se encarga de coordinar y ejecutar las operaciones sobre los datos; y un administrador de la base de datos, que es el profesional informático responsable de los aspectos técnicos y de seguridad de la base de datos.

"Una base de datos es un conjunto de archivos coordinados mediante una aplicación gestora en un espacio **físico definido y limitado**. Decimos que es un conjunto de archivos en forma genérica, es decir, son registros con datos respecto de una entidad porque cada modelo de base de datos llamará a este conjunto de registros de forma específica. Por ejemplo, el modelo relacional, que es uno de los modelos más difundidos de base de datos, los llama tablas, coordinados mediante una aplicación gestora llamada DBMS (Database Management System o sistema gestor de base de datos) que, en forma centralizada o distribuida, se encarga de coordinar y ejecutar todas las operaciones sobre los datos: ingreso, almacenamiento, recuperación, reporte, seguridad, etcétera, dentro de un espacio físico definido y limitado. Al instalarse la aplicación, este sistema gestor de base de datos se apropia de un espacio definido del disco rígido, dentro del cual almacenará todos los datos. Esto tiene una ventaja: una mayor facilidad y rapidez de recuperación; y una desventaja: la cantidad de datos que puede almacenar es limitada. De ahí que en el mercado hay productos diferentes que ofrecen diferentes capacidades de almacenamiento y manejo de datos."





El sistema gestor de la base de datos es, a su vez, un conjunto de aplicaciones. Veremos más adelante en detalle cada una de ellas, que se encarga básicamente de conectar y coordinar los requerimientos del usuario con los datos almacenados físicamente en la base de datos mediante un esquema de definición conceptual de la estructura de los registros.

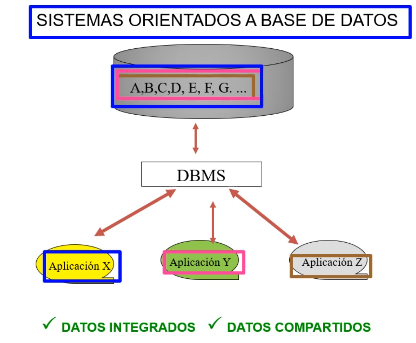


El sistema gestor de la base de datos es, a su vez, un conjunto de aplicaciones. Veremos más adelante en detalle cada una de ellas, que se encarga básicamente de conectar y coordinar los requerimientos del usuario con los datos almacenados físicamente en la base de datos mediante un esquema de definición conceptual de la estructura de los registros.

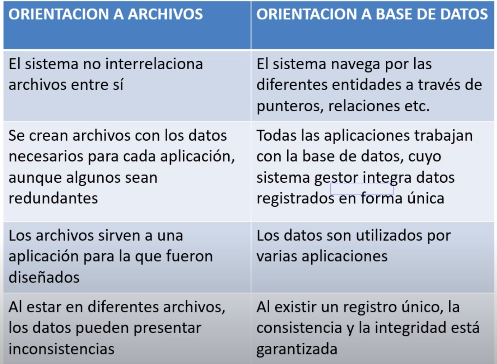


**Los primeros desarrollos de sistemas se hicieron utilizando archivos**. No existía todavía el concepto de base de datos, por eso los llamábamos sistemas orientados a archivos. Para cada aplicación se diseñaban archivos específicos cuyos registros podían contener atributos o campos similares. Por ejemplo, la aplicación de ABM de clientes utilizaba el archivo uno, que llamamos clientes, y que contenía registros con los siguientes atributos: el ID del cliente, el nombre, la dirección y la forma de pago. La aplicación de facturación utilizaba el archivo dos, llamado facturas, que contenía los siguientes atributos: el ID del cliente, dirección, número de factura y el importe. La aplicación de cobranzas utilizaba el archivo tres, llamado recibos, que contenía el ID de cliente, la forma de pago, importe y número de recibo.

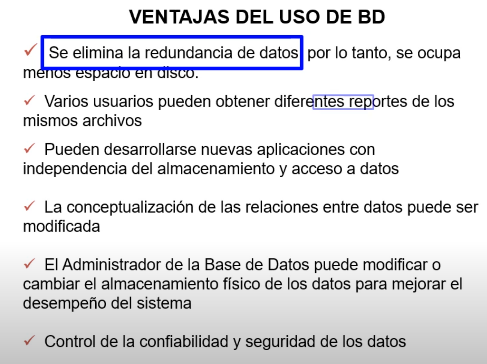
¿Cuáles eran los inconvenientes de estos sistemas orientados a archivos? Al tener datos repetidos, la consistencia entre ellos se volvía vulnerable. Por ejemplo, en el archivo uno, clientes, está registrada la dirección, y en el archivo dos, facturas, también. Si en la aplicación ABM de clientes se actualizaba la dirección, quedaba diferente e inconsistente con la dirección registrada en el archivo dos. Además, al estar este atributo duplicado, el volumen de almacenamiento aumentaba, es decir, el espacio de almacenamiento se volvía un problema. Por otra parte, el acceso a los datos de los archivos solo podía hacerse a través de las aplicaciones, y la estrategia de acceso que determinaba la velocidad y calidad de la recuperación de esos datos, etcétera, dependía totalmente de la habilidad y conocimiento del programador de la aplicación.



**Cuando se crea el modelo de sistemas orientados a bases de datos**, las aplicaciones a través del sistema gestor utilizan los mismos datos, que pasan a ser compartidos. Así, el problema de la redundancia que generaba inconsistencias se resuelve en las bases de datos de segunda generación, las relacionales, cuando aparece el concepto de integración de datos, eliminando el problema de la inconsistencia por duplicaciones mediante las técnicas de normalización. Un atributo de una entidad está registrado una y solo una vez en toda la base de datos.



Acá tenemos un cuadro resumen de las diferencias entre los sistemas orientados a archivos y los sistemas orientados a base de datos. A lo anteriormente dicho, agregamos que en los sistemas orientados a archivos, estos no se relacionaban entre sí; solo atendían a la aplicación para la que fueron creados. Mientras que en un sistema orientado a base de datos, los atributos de los archivos se pueden relacionar a través de punteros, relaciones, llaves, etcétera, según el modelo de base de datos.



El uso de bases de datos presenta las siguientes ventajas: se mejora el aprovechamiento del espacio en disco al eliminar las repeticiones de un mismo archivo o de varios; cada usuario puede seleccionar, según su necesidad, qué atributos requiere en un reporte; como los datos almacenados son compartidos por distintas aplicaciones con total independencia, se pueden crear nuevas aplicaciones que accedan a los mismos datos sin necesidad de modificar la estructura de los registros; el esquema de navegación mediante relaciones puede ser modificado sin alterar la estructura de los registros; el almacenamiento físico de los datos es transparente a las aplicaciones y a la vista de los usuarios, por lo tanto, el administrador de la base de datos puede hacer cambios en el almacenamiento físico que optimicen el desempeño del sistema gestor sin necesidad de modificar las aplicaciones; y el sistema gestor controla la confiabilidad y seguridad de los datos al no permitir el registro de datos redundantes y el acceso a usuarios no autorizados.

Dentro de una base de datos, los datos son almacenados siguiendo una estructura jerárquica que sigue un esquema descendente de lo general a lo particular.

En el nivel más alto, **una entidad representa** una abstracción del mundo real de la que puede guardarse información, por ejemplo, un empleado o un pedido de cliente

**Un archivo** es un conjunto de registros con la información almacenada de las instancias de una entidad; en nuestro ejemplo, de cada uno de los empleados se almacenan sus datos, que llamamos arbitrariamente empleados.

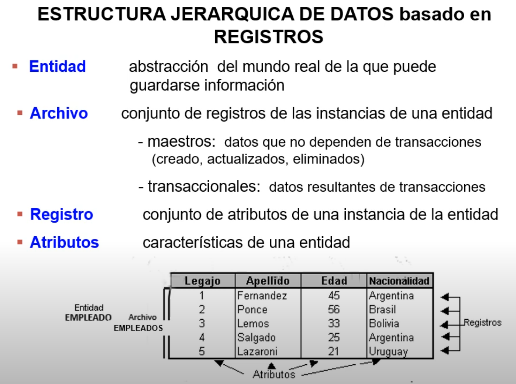
**Una instancia es** una ocurrencia específica de una entidad; en nuestro ejemplo, cada empleado es una instancia de la entidad empleado.

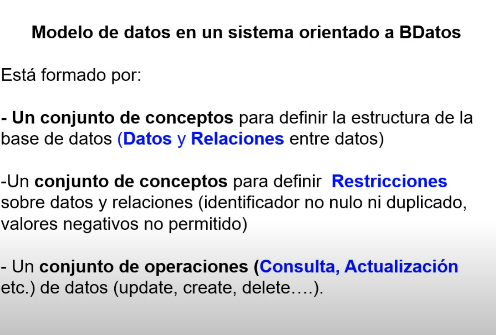
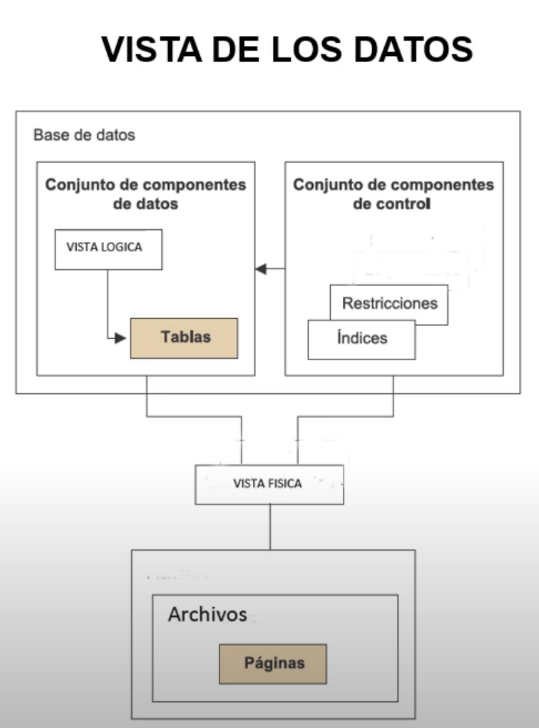
Los archivos pueden ser de tipo maestro, donde se registran los datos que no dependen de transacciones. Por ejemplo, un archivo de empleados registra los datos de cada uno de los empleados, independientemente de si ha trabajado o estado ausente. Otro ejemplo es un archivo de clientes, donde se registrarían sus datos fiscales, independientemente de los pedidos que haya hecho.

**También pueden ser de tipo transaccional,** que reflejan la actividad de una entidad; siguiendo nuestro ejemplo, sueldos o pedidos serían archivos transaccionales.

Veamos ahora el siguiente nivel**: un registro** es el conjunto de atributos de una instancia de una entidad.

**Los atributos** son las características que hemos definido como de interés para una entidad. Hay características de la entidad que no interesa registrar a los efectos de la aplicación o del negocio en cuestión. Por ejemplo, en el caso de empleados, no interesaría al negocio registrar los deportes que practica.



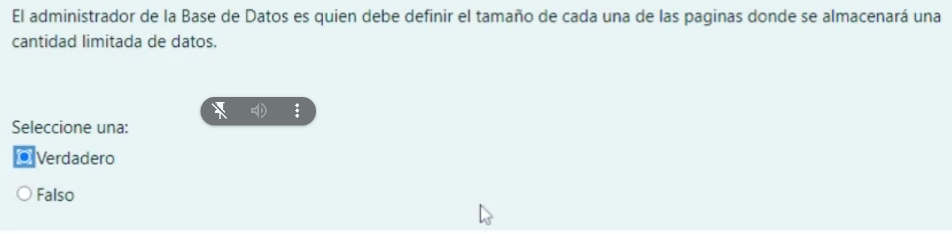


El modelo de datos estructurados en un sistema orientado a base de datos está conformado por los datos en sí, como unidad elemental, y por las relaciones que los conectan. Por ejemplo, un cliente ha efectuado determinados pedidos y ha hecho pagos de sus facturas. También se incluyen las restricciones que el sistema gestor impone para el registro de esos datos; por ejemplo, un identificador no puede ser nulo ni duplicado, una unidad de medida no puede ser negativa, y las operaciones permitidas sobre esos datos, como actualizaciones. Hay determinados atributos que tal vez no pueden ser modificados o eliminados.

Los datos almacenados en la base de datos tienen una vista lógica y una vista física.

**La vista lógica** describe los datos independientemente de la implementación física en el disco. Incluye las entidades, las relaciones entre ellas, los atributos de cada entidad, las características de cada uno de esos atributos y las restricciones sobre los datos y las relaciones. *Por ejemplo, en el modelo relacional, esta vista lógica se conceptualiza en tablas, y en el modelo columnar, en vectores*. Dentro del marco de restricciones y estrategias de acceso definidas por el modelo, los usuarios y los programadores pueden acceder a esta vista para solicitar reportes, organizar datos, programar aplicaciones que los utilizan, etcétera.

**En el nivel físico**, el sistema gestor de la base de datos utiliza las rutinas del sistema operativo para leer y escribir datos en el disco. El sistema operativo organiza los datos jerárquicamente en archivos y páginas, que son las unidades mínimas de entrada y salida. El administrador de la base de datos es quien debe definir el tamaño de cada una de las páginas, donde se almacenará una cantidad limitada de datos.



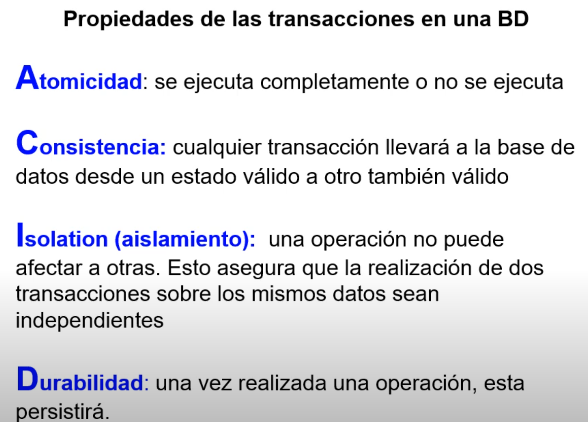
Un sistema gestor de base de datos, además de conceptualizar datos, relaciones y restricciones, puede **realizar transacciones** que deben cumplir ciertas propiedades.

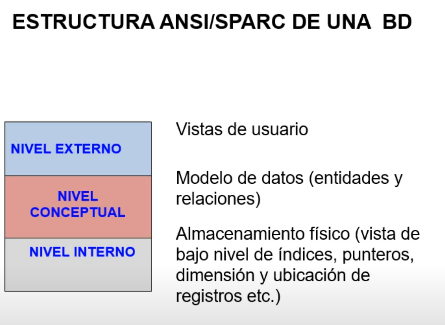
**Atomicidad:** Exige que una transacción debe ejecutarse en forma completa o no se ejecuta. Una transacción puede ser unitaria, es decir, contener una sola instrucción, o ser un conjunto de instrucciones que se ejecutan como una unidad indivisible o atómica. Cuando, por alguna causa o falla del sistema, no se puede ejecutar la totalidad de las instrucciones, se deshacen las instrucciones ya ejecutadas hasta dejar la base de datos en su estado original. *Por ejemplo, emitir una factura implica agregar un nuevo registro en las facturas y actualizar la deuda en el sistema.*

**Consistencia:** Cualquier transacción debe llevar la base de datos desde un estado válido a otro también válido, de acuerdo con las reglas establecidas, como el tipo de dato especificado o la unicidad de los registros respecto a las restricciones establecidas.

**Aislamiento:** Una operación no puede afectar a otras, asegurando que la realización de dos transacciones sobre los mismos datos sea independiente. Es decir, los estados intermedios de una transacción no deben ser visibles para la otra.

**Durabilidad**: Indica que, una vez realizada en forma completa una transacción, los datos persisten y no deben perderse bajo ninguna circunstancia.

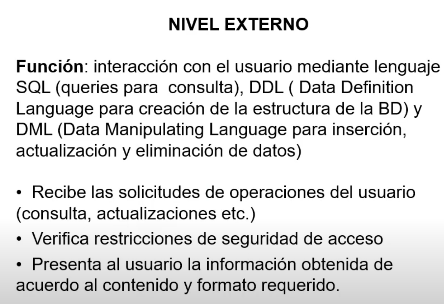




El modelo ANSI/SPARC es un estándar de diseño estructural de los sistemas de gestión de bases de datos. Se compone de tres niveles independientes que separan las vistas de los usuarios de alto nivel de los detalles físicos de almacenamiento en disco, ocultándolo a los usuarios. Este modelo consta de:

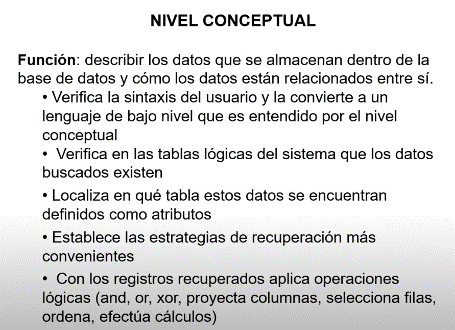
* **Nivel Externo o de Usuario:** Se relaciona con el nivel **interno o físico** mediante el **nivel conceptual** o lógico. En este nivel se describen los datos, sus relaciones, restricciones, etcétera, pero no se especifica cómo se almacenan físicamente.
* **Nivel Conceptual o Lógico:** Describe la estructura lógica de la base de datos, incluyendo las entidades, relaciones y restricciones, sin detallar cómo se almacenan físicamente en el disco.
* **Nivel Interno o Físico:** Se encarga de la implementación física de los datos en el almacenamiento. Es responsable de la organización de los datos en disco y de las técnicas de acceso y recuperación.

Veremos cada uno de estos niveles en detalle.



El nivel externo o de usuario interactúa con el usuario mediante lenguajes, como **SQL o SEQUEL,** para generar consultas (queries), **DDL** para la creación de la estructura de la base de datos, y **DML** para la inserción, actualización y eliminación de datos.

Sus funciones básicas son recibir las solicitudes de operaciones del usuario, como consultas y actualizaciones, verificar restricciones de seguridad de acceso de estos usuarios, y presentar al usuario el resultado de acuerdo al contenido y formato requerido en la solicitud.



El nivel conceptual se ocupa de describir los datos que se almacenan dentro de la base de datos y cómo esos datos están relacionados entre sí. Sus funciones son verificar la sintaxis del usuario y convertirla a un lenguaje de bajo nivel que es entendido por el nivel interno.

También verifica en las tablas lógicas del sistema que los datos buscados existan, localiza en qué tabla estos datos se encuentran definidos como atributos, establece las estrategias de recuperación más convenientes, y con los registros recuperados, aplica operaciones lógicas, como AND, OR, XOR, proyectar columnas, seleccionar filas, ordenar o efectuar cálculos.

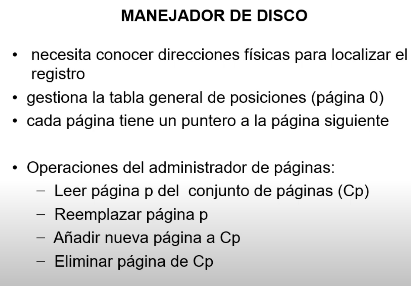


El nivel interno organiza los datos en el disco rígido de modo que el elemento requerido por el nivel conceptual se localice con un mínimo de entradas y salidas.

Recordemos que una página es la mínima unidad de entrada y salida manejada por el sistema operativo.

La **estructura de almacenamiento o diseño físico** de la base de datos se compone de dos aplicaciones:

1. el manejador de disco
2. el manejador de archivos.

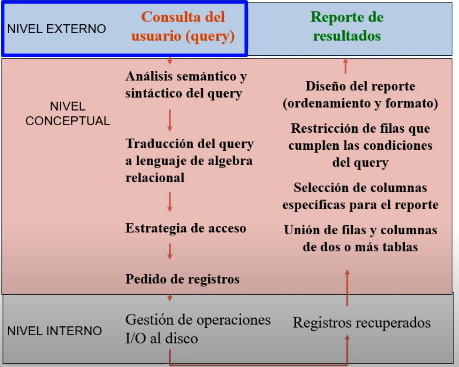
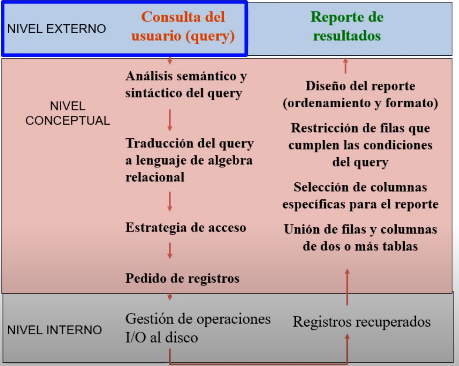


El **manejador de discos** necesita conocer las direcciones físicas para localizar el registro, y estas las obtiene gestionando la tabla general de posiciones, que vendría a ser la página cero. Cada página, a su vez, tiene un puntero a la página siguiente.

Las operaciones que puede administrar en cada página son: leer la página p del conjunto de páginas, reemplazar la página p, añadir una nueva página al conjunto de páginas, o eliminar una determinada página del conjunto de páginas.



**Con la dirección de páginas** enviada por el manejador de disco, el manejador de archivos puede leer un registro de ese archivo, añadir un registro al archivo, eliminar un registro del archivo, reemplazar un registro en el archivo, crear un nuevo archivo, o destruir un archivo.



Veamos cómo sería el proceso de pedido y emisión de un reporte.

Primero, un usuario del nivel externo, mediante una consulta en lenguaje SQL, solicita un reporte donde deben cumplirse determinadas condiciones de los registros, como transacciones de un determinado tipo dentro de un periodo de tiempo, etcétera.

Este requerimiento pasa al nivel conceptual, que verifica que la consulta o query es sintáctica y semánticamente correcta.

Como SQL es un lenguaje de alto nivel que no es interpretado por las siguientes instancias del sistema gestor, este nivel lo traduce a un lenguaje de álgebra relacional de bajo nivel.

El nivel conceptual recurre entonces al **diccionario de datos** para determinar en qué tablas se encuentran los atributos solicitados para el reporte y determina la estrategia de acceso a los registros,

solicitando su recuperación al nivel físico a través del sistema operativo.

El sistema operativo gestiona las operaciones de entrada y salida a las páginas del disco,

y con los registros recuperados se inicia el camino inverso: se devuelven los registros recuperados al nivel conceptual,

que procederá a unir las filas y columnas de las tablas recuperadas siguiendo el esquema de relaciones definidas en el diccionario de datos.

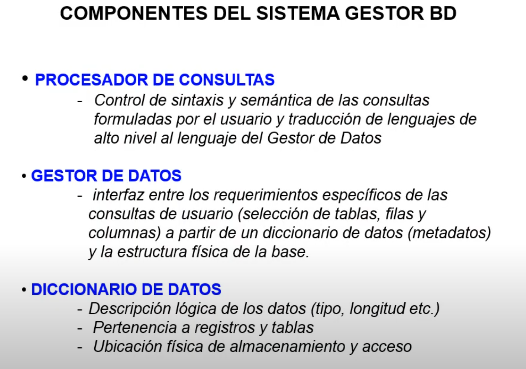
Selecciona las columnas de todos los atributos solicitados por el usuario, recordando que se recuperan registros completos que incluyen atributos que no son necesariamente requeridos en el reporte*. Por ejemplo, de un registro de clientes, solo podrían ser necesarios el nombre y la dirección*.

Restringe entonces los registros cuyos atributos cumplen las restricciones y condiciones exigidas por la consulta, como los clientes con domicilios en una determinada localidad.

Ordena las filas resultantes según el requerimiento del usuario, por ejemplo, ordenadas por localidad y dentro de estas por orden alfabético del nombre de los clientes.

Finalmente, el reporte es elevado al nivel externo y presentado a los usuarios, para quienes todo este proceso es transparente.

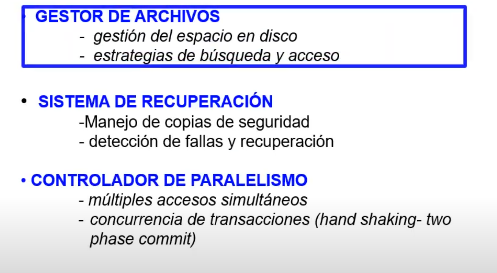
2.- El diccionario de datos es un listado de los datos (o campos) que componen un registro de la base de datos (por ej.: DNI, Nombre y apellido, dirección etc...), y detalles de cada uno (tipo de dato, longitud etc...)



Ya dijimos que un sistema gestor de base de datos es en realidad un conjunto de programas que realizan todas las operaciones necesarias para ingresar y recuperar los datos almacenados en una base de datos, como el proceso que acabamos de ver. Cada uno de estos programas tiene funciones específicas. Veamos qué realiza cada uno:

**Procesador de Consultas**: Recibe las consultas requeridas por un usuario autorizado, formuladas en un lenguaje de alto nivel como SQL. Controla la sintaxis (es decir, que las instrucciones estén correctamente escritas) y la semántica (que las consultas sean lógicas). Traduce la consulta a un lenguaje de bajo nivel que es interpretado por el sistema gestor.

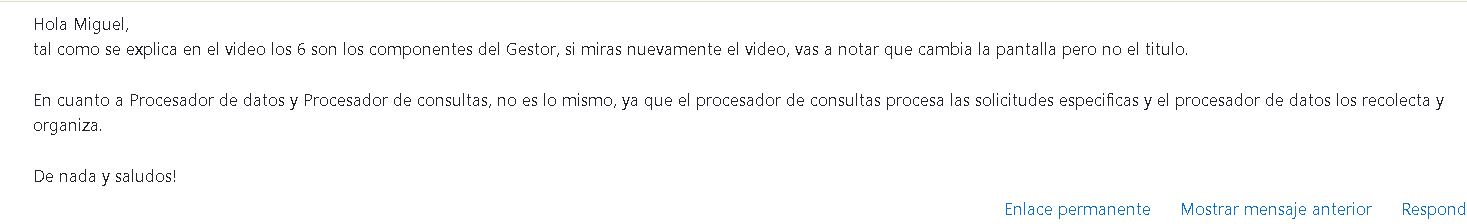
**Gestor de Datos:** Debe localizar las tablas donde se encuentran los atributos solicitados en la consulta. Esta información se encuentra en el **diccionario de datos.** Con esta información, solicita al gestor de archivos la recuperación de los registros. El gestor de archivos interactúa con el diccionario de datos, que es el encargado de gestionar toda la información lógica y física de los datos. Al recibir los registros recuperados desde el gestor de archivos, el gestor de datos procede a seleccionar las columnas y filas que cumplen con los criterios especificados en la consulta.

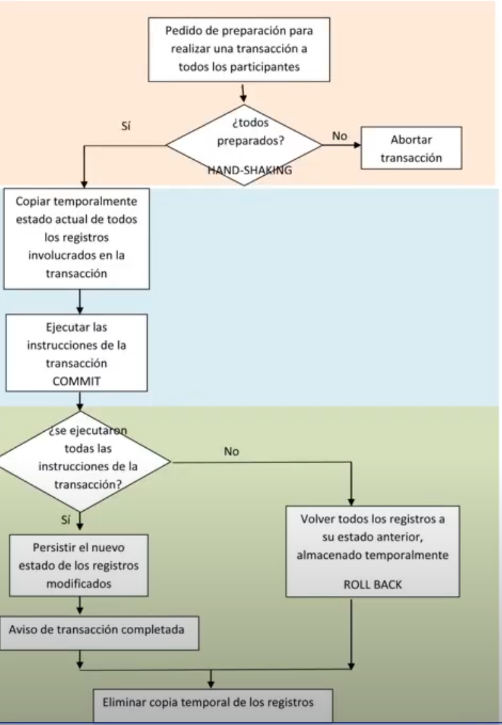


El **gestor de archivos**, encargado de gestionar el espacio en disco mediante páginas, establece la mejor estrategia de búsqueda y acceso para recuperar los registros solicitados.

S a todas estas aplicaciones, tenemos también el **sistema de recuperación**, que se encarga de detectar fallas en el procesamiento de transacciones, aplicando esquemas de guarda y recuperación definidos por el administrador de la base de datos.

El **programa controlador de paralelismo** ordena los accesos simultáneos para cumplir con las reglas ACID de atomicidad, consistencia, aislamiento y durabilidad. Manejando el protocolo llamado **handshaking** y **two-phase commit**, es decir, el protocolo de consenso y consolidación en dos fases.





Aquí tenemos un esquema del funcionamiento de este protocolo. Es un protocolo de consenso que permite que los nodos participantes de la transacción acuerden su disponibilidad para realizarla en forma integral o abortarla, dejando los registros que hayan realizado alguna modificación en su estado original.

*Recordemos el ejemplo de la emisión de una factura, donde deben ejecutarse las siguientes instrucciones para asegurar la atomicidad de la transacción: se debe calcular y registrar una nueva factura, actualizar la deuda del cliente en su cuenta corriente, y actualizar la existencia del producto en stock descontando las unidades vendidas. Están involucrados tres participantes: facturas, cuenta corriente del cliente y stock.*

El algoritmo básico tiene una fase de **preparación** /**petición** donde se envía un mensaje de consulta a todos los participantes de la transacción.

Cada participante responde con un mensaje de acuerdo o no.

Si alguno no acuerda, la transacción se cancela.

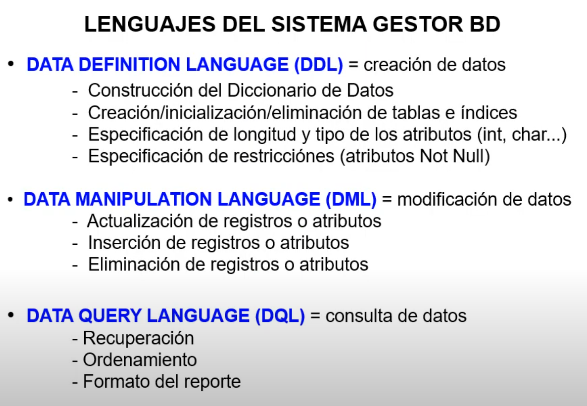
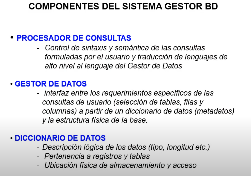
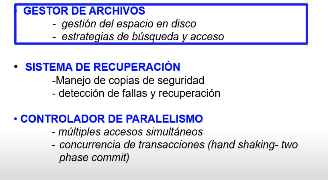
Cuando todos los participantes han acordado, se inicia la fase de consolidación (**commit**), en la que se bloquean los recursos involucrados y se copian los valores de los registros en forma temporal.

Cada participante recibe una instrucción para ejecutar su parte de la transacción y devuelve un mensaje de "ejecución realizada".

Cuando se han emitido los mensajes de ejecución de todos los participantes, la transacción se persiste y se avisa a los participantes para que desbloqueen los recursos.

Si no se reciben todos los mensajes de ejecución, la transacción se cancela, volviendo los registros que han sufrido alguna modificación a su estado anterior (**rollback**), de acuerdo al almacenamiento temporario realizado antes de iniciar la ejecución.

Finalmente, este registro temporario es eliminado.

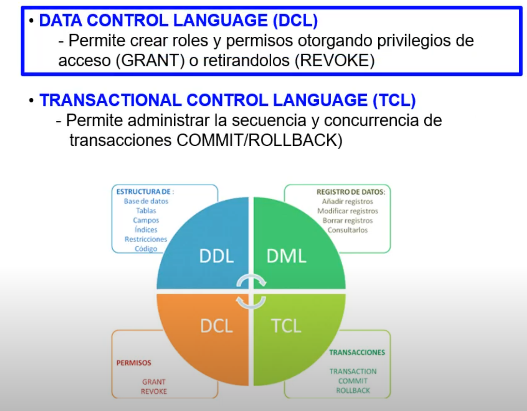


Las aplicaciones que mencionamos—procesador de consultas, gestor de datos, diccionario de datos, gestor de archivos y controlador de paralelismo—utilizan lenguajes específicos:

**DDL (Data Definition Language) o Lenguaje de Definición de Datos** permite a los programadores definir las estructuras que almacenarán los datos y permitirán consultarlos. Incluye la construcción del diccionario de datos, la creación y eliminación de tablas e índices, y la especificación de características y restricciones de los atributos.

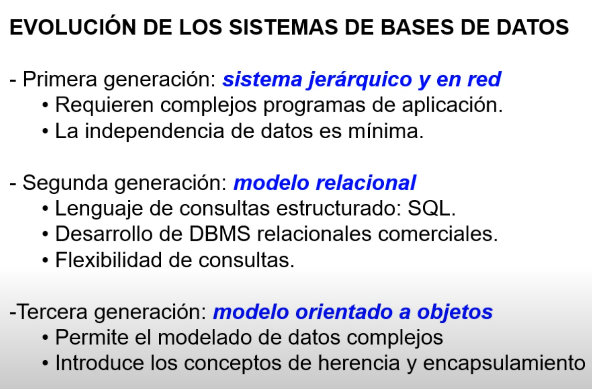
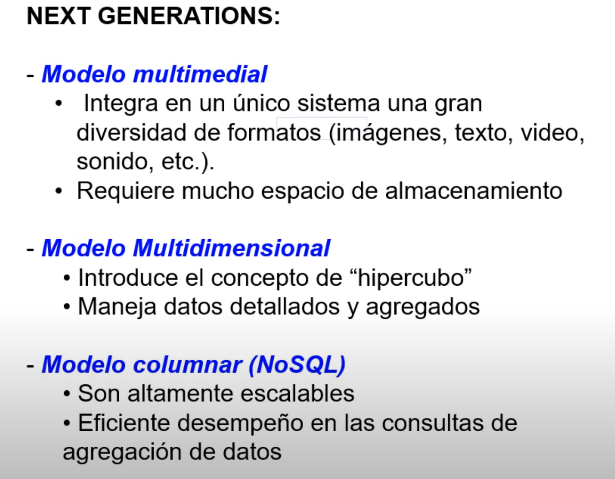
**DML (Data Manipulation Language) o Lenguaje de Manipulación de Datos** es el lenguaje que permite a los usuarios ingresar y modificar los datos.

**DQL (Data Query Language), más conocido como SQL (Structured Query Language),** permite a los usuarios especificar consultas de datos



**DCL (Data Control Language) o Lenguaje de Control de Datos** permite al administrador de la base de datos crear roles de usuario y concederles o revocar permisos de acceso.

**TCL (Transactional Control Language) o Lenguaje de Control de Transacciones** permite gestionar el protocolo de dos fases, que ya hemos detallado.



Las siguientes modelos en las nuevas generaciones en realidad no fueron numeradas y simplemente responden a las necesidades de manejo de datos que la evolución tecnológica generó.

**El modelo multimedial** es el que incorpora imágenes, textos, videos y sonido.

**El modelo multidimensional** maneja datos tanto detallados como agregados, usando un concepto de hipercubo.

**Y el modelo columnar o NoSQL** incorpora el manejo de datos y consultas por agregación. Son los que se utilizan actualmente para minería de datos.

Todos esos los vamos a ir viendo, como dijimos, uno por uno más adelante.

Ahora vamos a nombrar los modelos de bases de datos que fueron creados desde sus orígenes allá por la década del 70-80 hasta la actualidad para tener un panorama de su evolución. Más adelante estudiaremos cada uno en detalle.

**La primera generación utilizaba el modelo jerárquico y en red**. Eran complejos programas de acceso, la independencia de los datos era mínima. Por lo tanto, muy rápidamente se evolucionó a la

segunda generación, que **era el modelo relacional,** el más utilizado actualmente en el ámbito empresarial. Posibilita consultas dinámicas y flexibles en base al lenguaje query SQL.

**La tercera generación respondió a** la necesidad de datos complejos y resultó ser el modelo orientado a objetos, que se basa en los conceptos de herencia y encapsulamiento..

