Archivos	vision	**Archivo fisico:** es el archivo en la memoria secundaria y es administrado por el SO.
		Archivo logico: es el archivo utilizado desde el algoritmo. Cuando el algoritmo necesita operar con un archivo genera una conexión con el SO, el cual sera responsable de la administracion (independencia fisica).
	Tipo de accesos	**Secuencial:** el acceso a cada elemento de datos se realiza luego de haber accedido a su inmediato anterior.
		Secuencial indizado: el acceso a los elementos de un archivo se realiza teniendo presente algun tipo de organización previa sin tener en cuenta el orden fisico.
		Directo: es posibile recuperar un elemento de dato de un archivo con un solo acceso, conociendo sus caracteristicas, mas alla de su orden logico o fisico.
	Proceso de bajas	**Baja fisica:** consiste en borrar efectivamente la informacion del archivo, recuperando el espacio fisico. **Ventajas:** • Administra en cada mmento un archivo de datos que ocupa el lugar minimo necesario. **Desventajas:** • La perfomance. Ver a continuacion **Opciones:** • Generar un nuevo archivo con los elementos validos, es decir, sin copiar los que desea eliminar. (Un analisis basico determina que este metodo necesita leer tantos datos como tenga el archivo original y escribir todo los datos salvo el que se elimina y se debe disponer de memoria secundaria adicional para guardar el archivo viejo y el nuevo) • Utilizar el mismo archivo de datos, generando los reacomodamientos que sean necesarios. (Un analisis basico determina que la cantidad de lecturas a realizar es n, en tanto la cantidad de escrituras dependera del lugar donde se encuentre el elemento a borrar (peor caso n-1 escrituras) y no se requerie memoria adicional)
		Baja logica: consiste en borrar la informacion del archivo, pero sin recuperar el espacio fisico respectivo. **Ventajas:** • La perfomance, basta con localizar el registro a eliminar y marcarlo como no disponible. Por lo tanto se requiere tantas lecturas como sean necesarias hasta encontrar el elemento y una sola escritura para dejar la marca. **Desventaja:** • Al no recuperarse espacio en disco el tamaño del archivo tiende a crecer, por lo que se necesita emplear alguna tecnica alternativa que reutilize el espacio en desuso.

	Recuperar espacio (opciones)	**Recuperacion de espacio:** periodicamente utilizar el proceso de baja fisica para realizar un proceso de compactacion del archivo. Basicamente consiste en quitar todos los registros marcados como no disponibles.
		Reasignacion de espacio: otra alternativa posible consiste en recuperar el espacio, utilizando los lugares indicados como borrados para el ingreso (altas) de nuevos elementos al archivo.
	Reasignacion de espacio en reg. Longitud variable	**Primer ajuste:** consiste en seleccionar el primer espacio disponible donde quepa el registro a insertar. **Ventajas:** • Perfomance. Tiende a ser el mas rapido. **Mejor ajuste:** consiste en seleccionar el espacio mas adecuado para el registro. Se considera el espacio mas adecuado como aquel de menor tamaño donde quepa el registro. **Ventajas:** • Mas rapido que peor ajuste. **Peor ajuste:** consiste en seleccionar el espacio de mayor tamaño, asignado para el registro solo los bytes necesarios. **Ventajas** • Desde el punto de vista de fragmentacion generada. Peor ajuste mantiene la filosofia de trabajo de longitud variable, donde cada registro ocupa solamente el espacio que necesita.
	Fragmentacion al seleccionar espacio	**Fragmentacion interna:** aquella que se produce cuando a un elemento de dato se le asigna mayor espacio del necesario. **Pueden producirlo:** • Los registros de longitud fija. Se asigna tanto espacio como lo necesario de acuerdo con la definicion del tipo de datos. • Las tecnicas de primer y mejor ajuste. **Fragmentacion externa:** al espacio disponible entre dos registros, pero que es demasiado pequeño para poder ser reutilizado. **Pueden producirlo:** • Las tecnicas de peor ajuste.
	Busqueda binaria	Criterio de busqueda donde la mitad de los registros restantes se descartan en cada comparacion. **Ventaja:** • Mas rapido que la busqueda secuencial. **Desventaja:** • El archivo debe mantenerse ordenado y esto produce un alto costo adicional. • El archivo debe contener registros de longitud fija.

Indices	NRR	Numero relativo de un registro dentro de un archivo. Permite acceder directamente al registro deseado aunque es muy poco probable conocer el NRR del registro que contiene el dato a buscar.
	Ordenamiento basico en memoria principal	Para solucionar la perfomance al ordenar un archivo se plantea la solucion de ordenarlo en la memoria principal que es mucho mas rapida. Opciones: • Trasladar el archivo completo a memoria principal. **Desventaja:** ▪ Se debe tener la capacidad necesaria en memoria principal para trasladar el archivo. • Trasladar a memoria principal de cada registro cada clave que se utilize para ordenar junto a su respectivo NRR. **Ventaja:** ▪ Se aprovecha mejor el espacio en memoria principal **Desventaja:** ▪ Puede seguir faltando espacio en memoria principal para archivos muy grandes. • Dividir el archivo en particiones de igual tamaño, de modo tal que quepa en memoria principal. Enviar de a una particion a memoria principal (leyendo secuencialmente), ordenar las particiones alli y reescribirlas ordenadas (escritura secuencial) en memoria secundaria (estos pasos se llaman sort interno). Para finalizar se realiza un merge de las particiones. **Ventajas:** ▪ Produce particiones de igual tamaño haciendo el mergeo de particiones mas eficiente. Algoritmica simple. **Desventaja:** ▪ A mayor numero de particiones menor la perfomance final. • **Seleccion por remplazo:** Seleccionar siempre de memoria principal la clave menor, enviarla a memoria secundaria y remplazarla por una nueva clave que esta esperando ingresar a memoria principal. Seleccion natural: Es una variante de la selección por remplazo en donde utiliza memoria secundaria en la cual se insertan los registros no disponibles. **Ventajas:** ▪ Producen particiones con tamaño promedio igual o mayor al doble de la cantidad de registro, que caben en memoria principal. Dandonos al final menos cantidad de particiones que mergear mejorando la perfomance final. **Desventajas:** Tienden a generar muchos registros no disponibles. Las particiones no quedan

		necesariamente de igual tamaño. • Merge en mas de un paso...
	Indizacion	**Definicion:** Un indice es una estructura adicional que permite agilizar el acceso a la informacion almacenada en un archivo. En dicha estructura se almacenan las claves de los registros del archivo, junto a la referencia de acceso a cada registro asociado a la clave. **ES NECESARIO QUE LAS CLAVES PERMANESCAN ORDENADAS.** **Caracteristicas:** • Esta estructura es otro archivo de longitud fija, independiente de la estructura del archivo original. • Posibilita imponer orden en un archivo sin que realmente este se reacomode.
		Indice primario: Indice creado a partir de la clave primaria. **Operaciones:** • Creacion: Al crearse el archivo, se crea tabien el indice asociado, ambos vacios, solo con el registro encabezado. • Altas: Consiste simplemente en agregar el registro al final del archivo. A partir de esto con el NRR o la direccion del primer byte (según corresponda) + la clave primaria se genera un nuevo registro de datos a insertar en forma ordenada en el indice. Notar que si el indice se encuentra en memoria principal la insercion implica un costo relativamente bajo, mientras que en memoria secundaria el costo es mayor. COMO SIEMPRE! • Modificacion: Si considera la posibilidad de cambiar cualquier parte del registro excepto la clave primaria. Si el archivo esta organizado con registros de longitud fija el indice no se altera. Si el archivo esta organizado con registros de longitud variable y el registro modificado no cambia de longitud, nuevamente el indice no se altera. Si el registro modificado cambia la longitud, debe reubicarse en el archivo, cambiando asi su posicion la cual debe quedar asociada a la clave primaria en el indice. • Bajas: En este caso borrar un registro implicar borrar la informacion asociada en el indice primario. Por lo que se debe borrar fisica o logicamente el registro correspondiente en el indice. **Ventajas:** • Al ser de menor tamaño que el archivo asociado y tener registros de longitud fija, posibilita mejorar la perfomance de busqueda. • Se pueden realizar busqueda binarias, mientras que en el archivo original esto se ve condicionado a que tenga registros de longitud fija.
		Indices para Claves candidatas: Son claves que no admiten repeticiones de valores para sus atributos, similares a una clave primaria, pero que por cuestiones operativas no fueron seleccionadas como clave primaria. El tratamiento en operaciones es similar a las claves primarias.

		Indices secundarios: Son claves que soportan valores repetidos. Es una estructura adicional que permite relacionar una clave secundaria con una o mas claves. **Operaciones:** • Creacion: Similar al indice primario. • Alta: Cualquier alta genera una insercion en el indice secundario que implica reacomodar el archivo en el cual se almacena. Esta operación es de bajo costo en terminos de perfomance si el indice puede almacenarse en memoria principal pero resulta muy costosa en memoria secundaria. • Modificacion: Hay dos alternativas. La primera ocurre cuando se produce un cambio en la clave secundaria..en este caso se reacomoda el indice secundario. El segundo caso ocurre cuando cambia el resto del registro de datos (excepto clave primaria), no generando asi ningun cambio en el indice secundario. • Bajas: Esto implica que al eliminar un registro del archivo, se elimina la referencia en el indice primario, y todas las referencias que este tenga a indices secundarios. Se puede tambien eliminar solo la referencia al indice primario provocando una barrera de acceso a los indice secundarios los cuales no requeririan reordenarse, pero esto atenta a largo plazo con el espacio desperdiciado. **Alternativas de organización de indices secundarios:** • La primer alternativa implica almacenar en un mismo registro todas las ocurrencias de la misma clave secundaria. Esto consiste basicamente en que cada registro este formado por la clave secundaria + un arreglo de claves primarias correspondientes a dicha clave. El problema a esto es la eleccion de tamaño del registro, ya que conviene que este sea de longitud fija y puede terminar provando fragmentacion. • La segunda alternativa es pensar en una lista de claves primarias asociada a cada clave secundaria, a esto se lo denomina **LISTA INVERTIDA.** De esta manera no se debe realizar reserva de espacio y puede existir cualquier numero de claves primarias por cada clave secundaria. Para acceder a las claves primarias asociadas a la secundaria, primeramente se accede al archivo que conmtiene solo las claves secundarias, y alli se obtiene el NRR de acceso a la lista de claves primarias asociadas. Ambos archivos se organizan con registros de longitud fija. • Indices selectivos: Consiste en disponer de indices que incluyan solo claves asociadas a una parte de informacion existente, es decir, aquella informacio que tenga mayor interes de acceso. De esta forma, el indice incluye solo un subconjunto de datos asociado a los registros del archivo que interesa consultar, generando un filtro de acceso a la informacion de interes en el archivo.
	Arboles binarios	Un **arbol binario** es una estructura de datos dinamica no lineal, en la cual cada nodo puede tener a lo sumo dos hijos. Caracteristicas: • Tiene sentido cuando esta ordenado.

Arboles		• La busqueda en un arbol binario es de orden logaritmo si este esta balanceado. • Para poder utilizar los arboles como soporte de indices de busqueda, es necesario que los arboles binarios se implanten sobre almacenamiento secundario. **Ventajas:** • Representan una buena eleccion en terminos de insercion y borrado de elementos. **Desventaja:** • En terminos de busqueda siguen manteniendo un comportamiento similar al comportamiento obtenido al disponer de un archivo ordenado.
	Paginacion de arboles binarios	Son arboles que se dividen en paginas, es decir, se pagina, y cada pagina contiene un conjunto de nodos , los cuales estan ubicados en direcciones fisicas cercanas. Asi cuando se transfieren datos, no se accede al disco para transferir unos poco bytes, sino que se transmite una pagina completa. **Ventajas:** • Reduce el numero de accesos a disco necesarios para poder recuperar la informacion.
	Arbol multicamino	**Definicion:** Un arbol multicamino es una estructura de datos en la cual cada nodo puede contener k elementos y k+1 hijos. Un **arbol multicamino** representa otra forma de resolver el concepto de pagina que se vio anteriormente. **Desventaja:** • Continua manteniendo el problema de balanceo Se define al **orden** de un arbo multicamino como la cantidad maxima de descendientes posibles de un nodo.
	Arbol B	**Definicion:** Arboles multicamino con una construccion especial que permite mantenerlos balanceados a bajo costo. **Caracteristicas de un arbol b de orden M:** 1. Cada nodo del arbol puede contener, como maximo, M descendientes y M-1 elementos. 2. La raiz no posee descendientes directos o tiene al menos dos. 3. Un nodo con x descendientes directos contiene x-1 elementos 4. Los nodos terminales (hojas) tienen como minimo [M/2]-1 elementos y como maximo M-1 elementos. 5. Los nodos que no son terminales ni raiz tienen como minimo [M/2] descendientes. 6. Todos los nodos terminales se encuentran al mismo nivel. Mantiene el balanceo. **Ventajas:** • Cualquier operación (consulta, alta, baja o modificacion) se realiza en terminos aceptables de perfomance. **Desventaja:** • La perfomance en las operaciones se puede ver afectada a medida que H (altura del arbol) crece.

	Arbol B*	**Definicion:** Son una variante sobre los arboles B en donde se plantea y define una alternativa ante los casos de overflow. Asi, antes de dividir y generar nuevos nodos se dispone de una variante, redistribuir tambien ante una saturacion. Esta accion demorara la generacion de nuevos nodos y por ende tendra el efecto deaumenta en forma mas lenta la cantidad de niveles del arbol. Si la cantidad de niveles del arbol crece mas lentamente, la perfomance final de la estructura es mejor. **Caracteristicas:** 1. Cada nodo del arbol puede contener, como maximo, M descendientes y M-1 elementos. 2. La raiz no posee descendientes directos o tiene al menos dos. 3. Un nodo con x descendientes directos contiene x-1 elementos 4. Los nodos terminales (hojas) tienen como minimo [(2M-1)/3]-1 elementos y como maximo M-1 elementos. 5. Los nodos que no son terminales ni raiz tienen como minimo [(2M-1)/3] descendientes. 6. Todos los nodos terminales se encuentran al mismo nivel. Mantiene el balanceo.
	Arbol B+	**Definicion:** Es una estructura derivada de los arbols B y B* en la que incorpora ademas el tratamiento secuencial ordenado del archivo utilizando relacionando los nodos terminales mediante una **lista enlazada**. Asi se podra realizar busquedas aleatorias rapidas de informacion, en conjunto con acceso secuencial eficiente. **Caracteristicas:** 1. Cada nodo del arbol puede contener, como maximo, M descendientes y M-1 elementos. 2. La raiz no posee descendientes directos o tiene al menos dos. 3. Un nodo con x descendientes directos contiene x-1 elementos 4. Los nodos terminales (hojas) tienen como minimo [M/2]-1 elementos y como maximo M-1 elementos. 5. Los nodos que no son terminales ni raiz tienen como minimo [M/2] descendientes. 6. Todos los nodos terminales se encuentran al mismo nivel. Mantiene el balanceo. 7. Los nodos terminales representan un conjunto de datos y son enlazados entre ellos. Para poder realizar el acceso secuencial ordenado a todos los registros del archivo, es necesario que cada elemento (clave asociada a un registro de datos) aparezca almacenado en un nodo terminal. Asi, los arboles B+ diferencian los elementos que constituyen datos de aquellos que son separadores. **Detalles:** • Los elementos se insertan en nodos terminales. Si se produce una saturacion, el nodo se divide y se promociona una copia (el separador) del menor de los elementos mayores, hacia el nodo padre. Si el padre no tiene espacio para contenerlo, se dividira nuevamente. • Se debe notar que, en caso de dividir un nodo no terminal, se debe promocionar hacia el padre el elemento en si y no una copia del mismo, es decir, solo ante la division de un nodo terminal se debe promocionar una copia.

	Arboles B+ prefijo simple	Es un arbol B+ donde los separadores estan representados por la minima expresion posible de la clave, que permita decidir si la busqueda se realiza hacia la izquierda o hacia la derecha. Ventaja: • Disminuye el tamaño del arbol, al tener separadores de menor tamaño.
	Conclusiones a tener en cuenta	• La mejor estructura de un archivo va a depender del archivo en si y del proposito de uso de dicho archivo. • La familia de arboles B tienen una gran aplicabilidad en situaciones en las que resulta necesario acceder a un archivo tanto para busquedas aleatorias eficientes como para acceso secuencial. • En particular los arboles B* mejoran la eficiencia de los arboles B, aumentando la cantidad minima de elementos en cada nodo y por consiguiente ayudando a disminuir la altura final del arbol. El costo que se paga por ello es que las operaciones de insercion resultan un poco mas lentas. • En los arboles B+ toda la informacion de las claves esta contenida en los nodos terminales. • Los arboles B, B* y B+ siempre estan completamente balanceados • Los arboles B+ ocupan más espacio que los árboles B (ya que algunas claves se encuentran más de una vez en el árbol) • El proceso de eliminacion de un arbol B+ es más simple que en los árboles B, porque todas las claves se encuentran en las páginas hojas.
Hash (dispersion)	Conceptual	**Un archivo directo o con acceso directo** es un archivo en el cual cualquier registro puede ser accedido sin acceder antes a otros registros, es decir, cualquier regisgtro puede ser accedido directamente a diferencia de un archivo serie en el que un registro esta disponible solo cuando el registro predecesor fue procesado. Los archivos directos son almacenados en disco en diferentes formas respecto de los **archivos serie**. El mecanismo que trata de asegurar una recuperacion rapida de registros en un solo acceso promedio, lleva el nombre de dispersion o hashing.
	Que es	Hashing o dispersion es un metodo que mejora la eficiencia obtenida con arboles balanceados, asegurando en promedio un acceso para recuperar la información. Es un metodo en la que una clave se convierte en un numero, sobre el cual puede aplicarse la funcion hash, que permite obtener la direccion donde el registro se almacena dentro del archivo. **Atributos:** • No se requiere almacenamiento adicional. (aunque Hash asistido por tabla no lo cumple) • Facilita la insercion y eliminacion rapida de registros en el archivo. En general con un solo acceso a disco puede ser insertado o eliminado. Ventaja principal de utilizar este metodo de organización. • Localiza registros dentro del archivo con un solo acceso a disco. Si bien no es posible asegurar que cualquier registro sea encontrado en un solo acceso, la gran mayoria de las busquedas seran efectivamente

		satisfechas con un solo acceso a disco. **Limitaciones:** • No es posible aplicar la tecnica de dispersion en archivos con registros de longitud variable. • No es posible obtener un orden fisico de los datos. • No es posible tratar con claves duplicadas. Asi, no es aplicable la funcion hash sobre una clave secundaria. Uno puede aceptar trabajar con claves secundarias pero no cumpliria esta propiedad.
	Tipos de dispersion	**Hashing con espacio de direccionamiento estatico:** Es una politica donde el espacio disponible para dispersar los registros de un archivo de datos esta fijado previamente. Asi la funcion de hash aplicada a una clave da como resultado una direccion fisica posible dentro del espacio disponible para el archivo. **Hashing con espacio de direccionamiento dinamico:** Es una politica donde el espacio aumenta o disminuye en funcion de las necesidades de espacio que en cada momento tiene el archivo. Asi la funcion de hash aplicada a una clave da como resultado un valor intermedio, que sera utilizado para obtener una direccion fisica posible para el archivo. Estas direcciones fisicas no estan establecidas a priori y son generadas dinamicamente.
	Parametros de dispersion utilizando espacio de direcionamiento estatico	**Funcion de hash:** es una funcion que transforma un valor, que representa una llave primaria de un registro, en otro valor dentro de un determinado rango, que se utiliza como direccion fisica de acceso para insertar un registro en un archivo de datos. Puede generar colisiones o desbordes. Soluciones contra las colisiones: • Elegir un algoritmo de dispersion perfecto, que no genere colisiones. En general no es una opcion valida. • Minimizar el numero de colisiones a una cantidad aceptable y de esta manera tratar dichas colisiones como una condicion excepcional. Como disminuir el numero de colisiones: • Utilizar mas espacio en disco. La desventaja es basicamente el desperdicio de espacio. • Ubicar o almacenar mas de un registro por cada direccion fisica en el archivo. Que cada direccion obtenida por hash sea la direccion de un nodo o sector del disco donde es posible almacenar mas de un registro. Pero el nodo tiene un limite de espacio (previamente fijado) y si un registro ya no cabe en el nodo asignado se dice que el nodo esta satudaro o en overflow. **Tamaño de cada nodo de almacenamiento:** Es de lo que se hablaba en el parrafo anterior. Para determinar la capacidad del nodo queda determinada por la posibilidad de transferencia de informacion en cada operación e/s desde RAM hacia disco y viceversa. **Densidad de empaquetamiento (DE):** Es la relacion entre el espacio disponible para el archivo de datos y la cantidad de registros que integran dicho archivo. Cuanto mayor sea la DE, mayor sera la posibilidad de colisiones, dado que existe menos espacio de direcciones para dispersar registros. Cuando la DE se mantiene baja, se desperdicia espacio en el disco, pero la probabilidad de colisiones disminuye.

		Metodos de tratamiento de desbordes: Existen metodos para tratar los desbordes que ocurre cuando un registro es direccionado a un nodo que no dispone de capacidad para almacenarlo. Estos metodos deben cumplir dos objetivos: • Encontrar lugar para el registro en otra direccion. • Asegurarse de que el registro posteriormente sea encontrado.
	Resolucion de colisiones con overflow	**Saturacion progresiva:** El metodo consiste en almacenar el registro en la direccion siguiente mas proxima al nodo donde se produce saturacion. Este metodo necesita marcar si una direccion estuvo completa anteriormente a fin de no impedir la busqueda potencial de registros. Desventaja: • El metodo podria requerir chequear todas las direcciones disponibles en un caso extremo, para poder localizar un registro. **Saturacion progresiva encadenada:** Es una variante de saturacion progresiva. La diferencia es que una vez que se encuentra una direccion con espacio disponible esta se encadena o enlaza con la direccion base inicial, generando una cadena de busqueda de elementos. Ventajas: • Presenta mejoras de perfomance respecto a la saturacion progresiva. Desventaja: • Requiere que cada nodo manipule informacion extra: la direccion del nodo siguiente. **Doble dispersion:** El problema general que presenta la saturacion progresiva es que a medida que se producen saturaciones, los registros tienden a esparcirse en nodos cercanos. Esto puede producir **saturacion sectorizada.** Este metodo soluciona esto utilizando dos funciones hash. La primera obtiene a partir de la llave la direccion de base, en la cual el registro sera ubicado. De producirse overflow la segunda funcion hash no retorna una direccion sino que su resultado es un desplazamiento. Este desplazamiento se suma a la direccion base obtenida por la primera funcion, generando asi la nueva direccion donde se intentara ubicar el registo. En caso de haber overflow en la nueva direccion, se vuelva a sumar el desplazamiento hasta encontrar una direccion libre. Ventaja: • Tiende a esparcir los registros en saturacion a lo largo del archivo de datos. Desventaja: • Los registros en overflow tienden a ubicarse “lejos” de sus direcciones de base, lo cual produce un mayor desplazamiento de la cabeza lectora/grabadora del disco rigido, aumentando la latencia entre pistas y por consiguiente el tiempo de respuesta. **Area de desbordes por separado:** Ante la ocurrencia de overflow, los registros son dispersados en nodos que no se corresponden con su direccion base original. A medida que se completa el archivo por dispersion, puede suceder

		que muchos registros ocupen direcciones que no les corresponde pero que la operación de desborde lo alojo ahí. Para evitar esto, se sugiere como alternativa el uso de area de desbordes por separado. Entonces tendriamos nodos direccionables por la funcion hash y aquellos de reserva que solo podran ser utilizados en caso de saturacion pero que no son alcanzables por la funcion hash.
	Hash asistido por tabla	En los metodos anteriores a medida que se generar saturaciones, el numero de accesos requeridos aumenta. Hash asistido por tabla es una alternativa que utiliza **espacio de direccionamiento estatico,** soluciona esto y **asegura** acceder en un solo acceso a un registro de datos. Y no cumple con una propiedad mencionada anteriormente “no requiere almacenamiento adicional”; necesita una estructura adicional. **En que consiste:** El metodo utiliza tres funciones de hash: la primera de ellas retorna la direccion fisica del nodo donde el registro deberia residir; la segunda retorna un desplazamiento, similar al metodo de doble dispersion, y la tercera retorna una secuencia de bits que no pueden ser todos unos. **Caracteristicas:** • EL metodo asegura encontrar la clave buscada en un solo acceso a disco. • El costo asociado presenta dos aristas. La primera tiene que ver con la necesida de utilizar espacio extra para la administracion de la tabla en memoria principal. La segunda, en tanto, tiene que ver con la complejidad adicional en caso de una insercion que produsca saturacion. **Ventaja:** Este metodo pondera la busqueda sobre las otras operaciones.
	Hash extensible (usa espacio de direcciones dinamico)	El metodo de hash extensible consiste en comenzar a trabajar con un unico nodo para almacenar registros e ir aumentando la cantidad de direcciones disponibles a medida que los nodos se completan. Este metodo, al igual que el resto de los que trabajan con espacio dinamico, no utiliza el concepto DE. Caracteristicas: • El metodo necesita, para su implementacion, de una **estructura auxiliar**. Esta es una tabla que se administra en memoria principal. A diferencia del hash asistido por tabla, esta estructura contiene la direccion fisica de cada nodo. • **La funcion hash no retorna una direccion fisica**, sino una secuencia de bits. Dicha secuencia permite obtener de la tabla en memoria la direccion fisica del nodo para almacenar la llave. Ademas la tabla sera utilizada posteriormente para recuperar cada registro en un solo acceso a disco. • El proceso de busqueda asegura encontrar cada registro en un solo acceso. En caso de no encontrar el registro en dicho nodo, el elemento no forma parte del archivo de datos. Desventaja: En este y como en todos los metodos dinamicos la direcciones de nodos no estan prefijadas a priori y por lo tanto la funcion hash no puede retornar una direccion fija.

	Conclusion	Los metodos de busqueda de informacion en un archivo **presentan ventajas y desventajas en cada caso.** Como se discutio a lo largo de esta seccion de archivos, la busqueda secuencial de informacion tiene una perfomance de busqueda muy deficiente, pero el archivo generado no necesita mayor analisis en cuanto al proceso de altas o bajas. Tambien se indico oportunamente que el mayor porcentaje de operaciones sobre los archivos son de consultas; por lo tanto es necesario en una primera etapa maximizar la eficiencia del proceso de busqueda. Por este motivo se analizaron alternativas que contemplan generacion de indices y su implantacion con arboles. Los arboles balanceados son una solucion aceptable en terminos de eficiencia, reduciendo el numero de accesos a disco. Los B y B* tienen el mismo comportamiento, pero se diferencian en que **el B* completa mas los nodos.** La variante B+ presenta una ventaja adicional ya que permite no solo la busqueda eficiente sino tambien el recorrido secuencial del archivo. Pero cuando la eficiencia de busqueda debe ser extrema y se debe realizar un solo acceso a disco para recuperar la informacion, se dispone de la dispersion. La dispersion con **espacio de direccionamiento estatico asegura (en un gran porcentaje de los casos)** encontrar los registros buscados con un solo acceso a disco. Si se desea **asegurar siempre** un acceso a disco para recuperar la informacion la variante hash extensible que utiliza espacio de direccionamiento dinamico lo logra. **El costo que debe asumirse con esta tecnica es mayor procesamiento cuando una insercion genera overflow.**
Modelo	Abstracciones	**Abstraccion de clasificacion:** Se utiliza para definir una clase. Una clase es una declaracion o abstraccion de objetos, lo que significa que es la definicion de un objeto. **Abstraccion de agregacion:** Define una nueva clase a partir de una conjunto de otras clases que representan sus partes componentes. **Abstraccion de generalizacion:** Define una relacion de subconjunto entre los elementos de dos o mas clases.
	Propiedades de correspondencia entre clases	**Agregacion binaria:** Es la correspondencia existente entre dos clases diferentes. Ej: nacido_en es una agregacion binaria entre persona y ciudad. A partir de esa correspondencia puede definirse un nuevo concepto que es la cardinalidad entre las clases.

		Cardinalidad: Es el nivel de correspondencia existente entre los objetos de la clase. Se define el nivel minimo de correspondencia y el nivel maximo. Ejemplo: (nivel minimo, nivel maximo) persona → ciudad (1,1) porque la persona debe tener una ciudad donde nacio y porque no puede ser mas de una esa ciudad. Ciudad → persona (0,n) porque esa ciudad puede no tener ninguna persona nacida alli y puede tener una o mas. **Cobertura de generalizacion:** La propiedad de cobertura define el graado de relacion entre padres e hijos. Existen dos coberturas a definir. • **Total o Parcial:** Determina la relacion entre padres e hijos. Una cobertura es total cuando cada elemento del padre esta contenido en alguno de los hijos. La cobertura es parcial cuando pueden existir elementos padre que no se instancien sobre los hijos. • **Exclusiva o superpuesta:** Determina si un elemento del padre puede o no estar en mas de un hijo. Si solo puede estar en un hijo, la cobertura es exclusiva, en tanto que si puede estar en varios, se denomina superpuesta.
	Etapas	1. **Modelo conceptual**: Es desarrollado durante la etapa de adquisicion de conocimiento del problema; el analista se independiza del tipo de SGBD a utilizar y por consiguiente, del producto de mercado. Asi, el modelo conceptual se desarrolla independientemente de su implementacion final (relacional, de red, jerarquico o OO) 2. **Modelo logico:** El analista debe determinar el **tipo** de SGBD, debido a que las desiciones que debe tomar dependen de esa eleccion. 3. **Modelo fisico:** Es necesario tomar decisiones especificas. Estas ultimas tienen que ver con el producto de mercado a utilizar, es decir, el SGBD especificio.
Modelo conceptual	Caracterististicas	• **Expresividad,** es decir, capturar y presentar de la mejor forma la semantica de los datos del problema a resolver. • **Formalidad,** que requiere que cada elemento representado en el modelo sea perciso y bien definido, con una sola interpretacion posible. Esta formalidad es comparable con la formalidad matematica. • **Minimalidad,** caracteristica que establece que cada elemento del modelo conceptual tiene una unica formade representacion posible y no puede expresarse mediante otros conceptos. • **Simplicidad,** que establece que el modelo debe ser facil de entender por el cliente/usuario y el desarrollador.
	Componentes	**Entidades:** Representa un elemento u objeto del mundo real con identidad, es decir, se diferencia univocamente de cualquier otro objeto o cosa, incluso siendo del mismo tipo. **Conjunto de entidades:** Es una representacion que, a partir de las caracteristicas propias de cada entidad con propiedades comunes, se resume en un nucleo. **Relaciones:** Representan agregacion entre dos (binaria) o mas entidades. Describen las dependencias o

		asociaciones entre dichas entidades. **Conjunto de relaciones:** Es una representacion que a partir de las caracteristicas propias de cada relacion existente entre dos entidades, las resume en un nucleo. **Consideracion:** Cualquier problema puede ser resuelto con agregaciones binarias si se toman las decisiones de diseño correctas. **Atributos:** Un atributo representa una propiedad basica de una entidad o relacion. Es el equivalente a un campo de un registro. Para la entidad alumno, mediante los atributos se representan, por ejemplo, el nombre del alumno, su DNI o su numero de alumno. **Cardinalidad de los atributos:** Cuando se define un atributo se debe indicar si es o no obligatorio y si puede tomar mas de un valor (polivalente). • **Monovalente obligatorio:** En caso de que un atributo presente esta cardinalidad, no debe ser incluida la cardinalidad explicitamente en el modelo. • **Monovalente no obligatorio:**... • **Polivalente obligatorio:** Que puede existir multiples valores para este atributo (1..n) • **Polivalente no obligatorio: …**
	Componentes adicionales	**Jerarquias de generalizacion:** La generalizacion permite extraer propiedades comunes de varias entidades o relaciones y generar con ellas una superentidad que las aglutine. Asi, las caracteristicas compartidas son expresadas una unica vez en el modelo y los rasgos especificos de cada entidad quedan definidos en su subentidad. **La propiedad basica** de la abstraccion de generalizacion es la herencia. **Subconjuntos:** Los subconjuntos representan un caso especial de jerarquias de generalizacion. Hay problemas donde se tiene una generalizacion de la que se desprende solamente una especializacion. Este es el caso de los subconjuntos. La representacon es similar al caso anterior. Solamente hay que considerar que **no es necesario** indicar la cobertura para los subconjuntos. Esto se debe a que no puede tratarse de una cobertura total; si no, la especializacion y la generalizacion serian lo mismo, reperesantarian los mismos atributos. **Por lo tanto un subconjunto siempre indica cobertura parcial y exclusiva.** **Atributos compuestos:** Los atributos compuestos representan un atributo generado a partir de la combinacion de varios atributos simples. Un atributo compuesto podria ser polivalente o no obligatorio. Lo mismo podria ocurrir con los atributos simples que lo componen. **Indentificadores:** Es un atributo o un conjunto de atributos que permite reconocer o distinguir a una entidad de manera univoca dentro del conjunto de entidades. El cooncepto de identificador esta ligado a los conceptos de clave primaria y candidatas. En el modelo conceptual existen dos tipos: • **Simples o compuestos:** de acuerdo con la cantidad de atributos que lo conforman, el identificador es simple si esta conformado por solo un atributo y es compuesto en el resto de los casos.

		• **Internos o externos:** Si todos los atributos que conforman un identificador pertenecen a la entidad que identifica, es interno; en su defecto, es externo.
	Consideraciones	**Mecanismos de abstraccion**: • Los tres conceptos basicos (entidades, relaciones y atributos) se basan en la abstraccion de clasificacion • La abstraccion de agregacion esta presente en Entidades, Relaciones y atributos compuestos. • La abstraccion de generalizacion solo se ha presentado en entidades, **pero** es posible, ademas, utilizar la generalizacion de relaciones (aunque es muy poco frecuente)
Modelo entidad relacion logico	Caracteristicas	**Esquema conceptual:** El esquema logico debe representar la misma informacion disponible en el esquema conceptual. **Descripcion del modelo logico a obtener:** Aqui se deben definir las reglas que se aplicaran en el proceso de conversion. Esas reglas estan ligadas al tipo de SGBD seleccionado. **Criterios de rendimiendo de la BD:** durante la fase del diseño conceptual, se consideraron los requerimientos del usuario. No obstante, hay otro tipo de necesidades que no se pueden definir en el modelo conceptual. Estas necesidades tienen que ver con los requerimientos en general no funcionales del problema, como por ejemplo perfomance de la BD. **Informacion de carga de la BD:** Este concepto aparece, en cierta forma, ligado al concepto anterior. Cuando se genera el esquema logico, el analista debe observar cada entidad e interrelacion definida, y ver la probable evolucion de la informacion contenida en esas estructuras. De este modo, la desicion final sobre el esquema de una relacion o entidad dependera del numero probable de elementos que la compondran, con el proposito de mantener la perfomance bajo control.
	Desiciones sobre el diseño logico	**Atributos derivados:** Un atributo derivado si contiene informacion que puede obtenerse de otra forma desde el modelo. Es importante detectar dichos atributos y en el diseño logico se debe tomar la decision respecto de dejarlos o no. La ventaja de un atributo derivado es basicamente la disponibilidad de la informacion. Si la informacion que brinda ese atributo es **muy requerida** estara disponible rapidamente, sin la necesidad de calcular la informacion cada vez que se accede al atributo. Pero la desventaja radica en que necesita ser recalculado cada vez que se modifica la informacion que contiene, puede provar inconsistencias de datos a largo plazo. **Ciclos de relaciones:** Para mantener la minimalidad el diseño se debe eliminar los ciclos generadores en el modelo, es decir, se deben identificar las relaciones que generan repeticion innecesario de informacion e intentar obtener un modelo minimo o porque posteriormente el modelo implique menos tiempo de procesamiento. **Atributos polivalentes:** Los SGBD, en general, permiten que sus atributos contengan multiple valores. Pero este tipo de estructura es estatica, es decir, la cantidad de lugares previstes para almacenar informacion esta predeterminada. Ningun SGBD relacional permite que un atributo contenga valores multiples determinados

dinamicamente. En este caso la solucion debe implementarse con otro criterio. Este criterio se denomina primera forma normal.
Un modelo esta en primera formal normal (1FN) si todos los atributos de entidades o relaciones son atributos simples. Para obtener esto en el caso de los atributos polivalentes se debe quitar como atributo, y generar una nueva entidad y que se establesca una relacion con la entidad que tenia que el atributo polivalente.
Atributos compuestos: Son atributos que estan conformados por varios atributos simples. Los SGBD en general no soportan esta variante. Existen tres soluciones para tratar estos atributos:

1. La primer solucion consiste en generar un unico atributo que se convierta en la concatenacion de todos los atributos simples que contiene el atributo compuesto. Esta solucion es simple y sencilla pero al unir todos los atributos se pierde la identidad de cada atributo simple.
2. La segunda solucion plantea definir todos los atributos simples sin un atributo compuesto que los resuma. Esta solucion es, en general la mas indicada.
3. La tercer solucion presenta una alternativa radical. Consiste en generar una nueva entidad, la que representa el atributo compuesto, conformada por cada uno de los atributos simples que contiene. Esta nueva entidad debe estar relacionada con la entidad a la cual pertenecia el atributo compuesto. Es una opcion mas compleja de realizar pero capta mejor la esencia del atributo compuesto.

Jerarquias: Las decisiones para manipular las jerarquias en un modelo logico es el punto mas importante de convertir el modelo conceptual a logico.
El modelo relacional no permite jerarquias por lo que hay tres opciones para tratar una jerarquia:

1. Eliminar las especializaciones (subentidades o entidades hijas), dejando solo la generalizacion (entidad padre), la cual debera incorporar todos los atributos de sus hijos **y es necesario que cada uno de estos atributos sean opcionales.**
2. Eliminar la entidad generalizacion (entidad padre), dejando solo las especializaciones. Con esta solucion, los atributos del padre deberan incorporarse en cada uno de los hijos.
3. Dejar todas las entidades de la jerarquia, convirtiendola en relaciones (1,1) entre el padre y cada uno de los hijos. Esta solucion pemite que las entidades que conforman la jerarquia mantengan sus atributos originales, generando la relacion explicita ES_UN entre padre e hijos.

Excepciones: Si la cobertura de la jerarquia fuese parcial, la segunda solucion no es aplicable. Una cobertura parcial significa que algunos elementos contenidos en la entidad padre no estan cubierto por las hijas. Esto implica, que si se quita al padre, dichos elementos se pierden. Esto provoca que se pierda informacion por lo que no es aplicable.
Si la cobertura es superpuesta, la segunda solucion nuevamente es poco practica. Algunos elementos del padre se repiten en varios hijos; esto significa que debera repetir informacion en la subentiendades generadas. SI bien esto no representa un problema en si mismo, la repeticion innecesaria de informacion puede ocacionar

		inconvenientes cuando se utilice la BD generada. **Conclusion:** Se puede afirmar que la tercera alternativa de solucion es la que capta mejor la esencia de la herencia y por ende la que resulta mas interesante aplicar. Sin embargo, esta solucion es la genera mayor numero de entidades y relaciones en el modelo final. Esto podria significar a futuro problemas de perfomance en la utilizacion de la BD. Los **subconjuntos,** caso especial de jerarquias, tienen una cobertura parcial exclusiva. En este caso, tanto la primera como la tercera alternativa son aplicables, aunque la segunda es descartada. **Particion de entidades:** El motivo de partir una entidad es reorganizar la distribucion de las entidades en un conjunto de entidades que la componen (particion horizontal) o de los atributos que conforman cada conjunto de entidades (particion vertical). Ejemplo de particion horizontal: Un conjunto de entidades persona que agrupa tanto a los clientes como a los proveedores de cierta organización se puede partir en dos entidades. Una clientes y otra proveedores. Ejemplo de particion vertical: Una entidad persona que tiene DNI, apellido, horas trabajadas, salario, obra social, titulo y habilidades laborales como atributos se puede partir verticalmente resultando en 3 entidades, una llamada datos_personales, datos_laborales y datos_salariales. Se debe especificar un atributo que permite definir la relacion entre las 3 nuevas entidades. **Objetivos:** • Mejorar la perfomance de las operaciones futuras sobre la BD. • Mejorar los niveles de seguridad de la BD, para lo cual se otorgan determinados derechos de acceso a los usuarios de la BD. **Transformacion de minimalidad:** Un esquema es minimo cuando una idea se expresa en el una sola vez. La redundancia de informacion puede ser un efecto positivo si es conocida y controlada, o un efecto negativo si aparece en el esquema por deficiencia en la modelizacion. Los **atributos derivados** representan la primera causa que afecta la minimalidad. Un atributo derivado es un atributo que aparece en el modelo, pero cuya informacion, si no existiera almacenada en el, podria igualmente ser obtenida. Los **ciclos de relaciones** presentan la segunda causa que afecta la minimalidad. Se dice que existe un ciclo cuando una entidad A esta relacionada con entidad B, la cual esta relacionada con una entidad C, la que a su vez se relaciona con la entidad A.
Modelo fisico (relacional)	Definicion	EL modelo relacional representa a una BD como una colección de archivos denominados tablas, las cuales se conforman por registros. Cada tabla se denomina relacion y eta integrada por filas horizontales y columnas verticales. Cada fila representa un registro del archivo y se denomina tupla, mientras que cada columna representa un atributo del registro.

		La definicion matematica de las relaciones se desarrolla a partir de la nocion de dominio. Un dominio representa un conjunto de posibles valores para un atributo. Como un dominio restringe los valores del atributo, puede considerarse como una restriccion.
	Pasos	**Eliminacion de identificadores externos:** El primer paso en la conversacion del esquema logico hacia el esquema fisico consiste en la eliminacion de los identificadores externos. Cada una de las entidades que conforman el esquema logico debe poseer sus identificadores definidos de forma interna. Para lograr esto, se deben incorporar, dentro de la entidad que contenga identificadores externos, aquellos atributos que permiten la definicion del identificador de forma interna a la entidad. **¿Que identificador se deberia seleccionar en caso de que haya mas de uno?:** Se debera elegir el identificador mas representativo, es decir, el que posteriormente sera definido como clave primaria. **Seleccion de claves: primaria, candidate y secundaria:** En el modelo fisico se debe decidir cuales de los identificadores se convierte en clave primaria o candidata. Si una entidad solo tiene definido un identificador, ese identificador es clave primaria de la tabla. Si la entidad tuviese definidos varios identificadores, la selección de la **Clave Primaria (CP)** deberia realizarse del siguiente modo: • Entre un identificador simple y uno compuesto deberia tomarse el simple dado que asi es mas facil de tratar y/o usar. • Entre dos identificadores simples, se debe optar por aquel de menor tamaño fisico. • Entre dos identificadores compuestos, se deberia optar por aquel que tenga menor tamaño de bytes. De ese modo, al construir un indice utilizando un arbol B como estructura, es posible almacenar mayor cantidad de claves por nodo. Las consideraciones anteriores definen, **en general**, el criterio mas adecuado para elegir la CP. El resto de los identificadores sera definido como **Clave Candidata (CC)**. Sin embargo, los SGBD ofrecen una alternativa que resulta ser la mas conveniente. Utilizar un atributo con dominion Autoincremental, asi una tabla con dicho atributo tiene definida una CP que es trata por el SGBD en forma exclusiva. El usuario solamente tiene permitida la operación de consulta sobre la CP, es decir, no la puede generar, borrar ni modificar. Esta elexccion, combinada con el concepto de integridad referencial genera una CP que actua de la forma mas eficiente posible, mejorando la perfomance final de la BD. **Concepto de super clave:** Una superclave es un conjunto de uno o mas atributos que permiten identificar de forma unica una entidad de un conjunto de entidades. Presentada de esta manera, una superclave es equivalente a una CP o una CC. Sin embargo, una superclave puede contener atributos inecesarios. Si un atributo es superclave, entonce tambien lo es cualquier superconjunto que incluya a dicho atributo. Una CP o CC es una superclave que no admite a un subconjunto de ella como superclave.

		Clave Foranea (CF): Se denomina clave foranea a un atributo o grupo de atributos de una tabla referida a un atributo o grupo de atributos que en otra tabla son CP. Sobre estos atributos se establece un concepto de integridad referencial. **Una CF es ademas clave secundaria en la tabla donde aparece.**
	Conversion de entidades	El proceso de conversion para obtener el esquema fisico de una BD comienza con el analisis de las entidades definidas en el modelo logico. En general, cada una de las entidades definidas se convierte en una tabla del modelo. **Existe una excepcion:** Cuando existe una relacion uno a uno con cobertura total entre dos entidades se debe generar una unica tabla que aglutine a los atributos de ambas.
	Conversion de relaciones. Cardinalidad	**Cardinalidad muchos a muchos:** Es la mas sencilla de implementar. La solucion propuesta es generar en total 3 tablas, las dos tablas que definidas como entidad en modelo logico + la relacion N a N convertida a tabla, conformada por los atributos que definen la CP de cada una de las entidades que relaciona. En este caso, ambos atributos generan la CP de la nueva tabla. **Cardinalidad de uno a muchos:** La decision debera ser tomada en funcion de la cardinalidad minima definida y de las decisiones de diseño. • **Uno a muchos con participacion total de ambos lados:** (ejemplo con personal y obra social) Se generan dos tablas y en la tabla personal se agrega una Clave foranea que es la CP de obra social. • **Uno a muchos con participacion parcial del lado de muchos:** (ejemplo personas y localidades) Se generan dos tablas y en la tabla personas se agrega una clave foranea que es la cp de localidades. • **Uno a muchos con participacion parcial del lado de uno:** (ejemplo personas y grupo_sanguineos) Como puede permir valores nulos aquellas personas que no conozcan su grupo sanguineo se puede generar 2 tablas como vimos anteriormente pero esto deriva en administrar posibles valores nulos (no recomendable) por lo que es mejor genera 3 tablas. PERSONAS, GRUPOS_SANGUINEOS Y CONOCE (similar a n a n). • **Uno a muchos con cobertura parcial de ambos lados:** Se tiene las mismas opciones que con Uno a muchos con participacion parcial del lado de uno. **Cardinelidad uno a uno:** • **Con Participacion es total de ambos lados:** se debe generar una sola tabla que contenga los atributos de ambas entidades. • **Con participacion parcial de un lado:** (ejemplo personal, empleados, ingenieros y directivos) Cada empleado, ingeniero y directivo es un personal y cada personal puede ser uno de estos tres casos. La solucion es definir la tabla PERSONAL y luego definir las tablas EMPLEADOS, INGENIEROS Y DIRECTIVOS en donde la CP de PERSONAL sera CP en estass ultimas 3 tablas. Es posible que se repita la misma CP. Se debe destacar que en estas ultimas 3 tablas el atributo CP no es autoincremental.

		• **Con participacion parcial de ambos lados:** No es habitual pero la solucion aconsejable en este caso es generar una tabla por cada entidad y otra tabla que involucre la relacion. Esta ultima tabla debera incluir las CP de las entidades que relaciona.
	Integridad Referencial	La **integridad referencial (IR)** es una propiedad deseable de las BD relacionales. Esta propiedad asegura que un valor que aparece para un atributo en una tabla aparezca ademas en otra tabla para el mismo atributo. La IR plantea restricciones entre tablas y sirve para mantener la consistencia entre las tuplas de dichas tablas. Cada SGBD tiene escenarios de definicion de IR diferentes, pero en general cuando se la define se puede optar por estas situaciones: • **Restringir la operación:** es decir, si se intenta borrar o modificar una tupla que tiene IR con otra, la operación se restringe y no se puede llevar a cabo.. • **Realizar la operación "en cascada":** en este caso, si se intenta borrar o modificar una tupla sobre la tabla donde esta definida la CP de la IR, la operación se realiza en cadena sobre todas las tuplas de la tabla que tiene definida la CF. • **Establecer la CF en nulo:** Si se borra o modifica el valor del atributo que es CP, sobre la CF se establece valor nulo. Esta opcion es muy utilizada ni esta presente en todos los SGBD. • **No hacer nada:** en este caso se le indica al SGBD que no es necesario controlar la IR. Esta opcion es equivalente a no definir restricciones de IR.
Normalizacion	Objetivo	Una BD debe normalizarse para evitar la redundancia de los datos, dado que cuando se repite innecesariamente la informacion, aumentan los costos de mantenerla actualizada. Ademas, la normalizacion ayuda a proteger la integridad de la informacion de la BD. Propiedades deseables: • Evitar redundancia de datos • Evitar anomalias de actualizacion • Evitar perdida de integridad de datos La ventaja de utilizar una BD normalizada consiste en disponer de tablas, cuyos datos seran para el usuario de facil acceso y sencillo mantenimiento.
	Redundancia y anomalias	Cuando se resuelve una resolucion de muchos a muchos del esquema logico, se genera una nueva tabla con las CP de las dos entidades que relacion: esta solucion repite informacion. Sin embargo, esta redundancia es necesaria para representar en el modelo fisico la relacion. Este tipo de redundancia se denomina rendundancia deseada y no afecta al proceso de normalizacion. Por el contrario, existen otros casos de redundancia, denonimadas no deseadas, que generan anomalias de actualizacion cuando se opera.

		• Anomalias de insercion • Anomalias de borrado • Anomalias de modificacion
	Dependencias funcionales	Una dependendencia funcional (DF) representa una restriccion entre atributos de una tabla de la BD. Se dice que un atributo Y depende funcionalmente de un atributo X (anotado como X → Y), cuando para un valor de x siempre se encuentra el mismo valor para el atributo Y. Entonces X (determinante) determina a Y (consecuente). Se puede notar que de un atributo que es CP se desprenden DF al resto de los atributos de la tabla. **Es decir, cualquier atributo depende funcionalmente de la CP. Algo similar ocurre con cualquier atributo que haya sido definido como CC. Por otro lado puede darse de que un atributo que es CC dependa de la CP y viceverza. Esta condicion solo se presenta entre un atributo que es CP y otro que es CC, o cuando ambos son CC.** **Caracteristicas:** • El consecuente de la DF debe estar formado por un solo atributo. Si se define la DF X → Y, Y debe ser solo un atributo. • Si se define la DF X → Y, no es posible encontrar otra dependencia Z → Y donde Z sea un subconjunto de atributos de X. En este caso se define a la DF como completa. • No se puede quitar de un conjunto de DF algunas de ellas, y seguir teniendo un conjunto equivalente.
	Primera forma normal	Un modelo esta en 1FN si todos los atributos que conforman las tablas del modelo son atributos monovalentes. Los atributos polivalentes generan inconvenientes para la definicion de una tabla. Solucion quitar el atributo polivalente creando una nueva entidad. Este paso se realizo en el esquema logico; para que luego al obtener el esquema fisico ya tuvieramos el modelo en 1FN
	Dependencia funcional parcial y 2FN	Una DF X → Y se denomina parcial cuando, ademas, existe otra dependencia Z → Y, siendo Z un subconjunto de X. PEDIDOS = (<u>idPedido, idProducto</u>, descripcionproducto, fechapedido, cantidad) DFs: idPedido, idproducto → descripcionproducto idPedido, idproducto → fechapedido idPedido, idproducto → cantidad DFs parciales: idProducto → descripcionproducto

		idPedido → fechapedido Un modelo esta en segunda forma normal (2FN) si esta en 1FN y ademas no existe en ninguna tabla del modelo una DF parcial. Solucion al ejemplo anterior: DETALLESPEDIDOS = (idPedido, idProducto, cantidad) PEDIDOS = (idPedido, fechapedido) PRODUCTOS = (idProducto, descripcionProducto)
	Dependencia funcional transitiva y 3FN	Una DF X → Y se denomina transitiva cuando existe un atributo Z tal que X → Z y Z ->Y. ALUMNO = (idAlumno, nombre, direccion, idcarrera, nombre_carrera) DFs: idalumno → nombre idalumno → direccion idalumno → idcarrera idalumno → nombre_carrera DFs transitiva: **Notamos tambien** que existe idcarrera → nombre_carrera **Entonces** idalumno → nombre_carrera produce una DF transitiva. nombre_carrera es informacion repetida que podria obtenerse a partir del idcarrera. Un modelo esta en tercera forma normal (3FN) si esta en 2FN y ademas, no existe en ninguna tabla del modelo una DF transitiva. Solucion al ejemplo CARRERA = (idCarrera, nombre_carrera) ALUMNO = (idAlumno, nombre, direccion, idcarrera)
	Dependencia funcional de Boyce-codd y forma normal	La DF de Boyce-Codd esta basada en DF que tienen en cuenta las CC de una tabla. Una DF X → Y se denominca Dependencia funcional de Boyce-codd (DFBC) cuando X no es una CP o CC, e Y es una CP o CC, o parte de ella.

	de Boyce-codd	DICTA = (nombre_alumno, nombre_materia, nombre_docente) CC = (nombre_alumno, nombre_materia) y (nombre_alumno, nombre_docente). DFs: nombre_alumno, nombre_materia → nombre_docente nombre_alumno, nombre_docente → nombre_materia Pero existe una tercera DF: nombre_docente → nombre_materia **dado que un docente solamente puede dictar una materia.** Esta ultima genera una DF de BC, el determinante no es CC ni CP y el consecuente (nombre_materia) es parte de una CC. Un modelo esta en Forma Normal de Boyce-Codd (FNBC, mas conocida por BCNF) si esta en 3FN y ademas no existe en ninguna tabla del modelo una DF de BC. Solucion al ejemplo: DICTA = (nombre_docente, nombre_materia) CURSA = (nombre_alumno, nombre_materia)
	Dependencia multivaluada y cuarta forma normal (4FN)	Existen otro tipo de restricciones que plantean dependencias que no pueden ser consideradas como DF. Estas nuevas dependencias se denominan multivaluadas. Una dependencia multivaluada (DM) denotada como X → → Y, siendo X e Y conjuntos de atributos en una tabla, indica que para un valor determinado de X es posible determinar multiples valores para el atributo Y. Un modelo se encuentra en 4FN si esta en FNBC y las DM que se puedan encontrar son triviales. Una DM X → → Y se considera trivial si Y esta multideterminada por el atributo X y por un subconjunto de X Dada la DM (X, Z) → → Y, esta sera trivial si Y esta multideterminado unicamente por el par (X, Z). Si Y esta multideterminado solamente por X o Z, la dependencia no se considera trivial.
	Quinta forma normal	Es dificil que lo tomen.
	A tener en cuenta	Recordar que el proceso de normalizacion no es estrictamente obligatorio. La normalizacion evita redundancia de datos, pero al mismo tiempo segmenta mas la informacion.

Conceptos de transacciones	Conceptos	Una transaccion es un conjunto de instrucciones que actua como unidad logica de trabajo. Esto es, una trasaccion se completa cuando se ejecutaron todas las instrucciones que la componen. En caso de no poder ejecutar todas las instrucciones, ninguna de ellas debe llevarse a cabo. Para garantizar que una transaccion mantenga la consistencia de la BD es necesario que cumpla con cuatro propiedades basicas: • **Atomicidad:** garantiza que todas las instrucciones de una transaccion se ejecutan sobre la BD, o ninguna de ellas se lleva a cabo. • **Consistencia:** la ejecucion completa de una transaccion lleva a la BD de un estado consistente a otro estado de consistencia. Para esto la transaccion debe estar escrita correctamente. • **Aislamiento:** cada transaccion actua como unica en el sistema. Esto significa que la ejecucion de una transaccion, T1, no debe afectar la ejecucion simultanea de otra transaccion, T2. • **Durabilidad:** una vez finalizada una transaccion, los efectos producidos en la BD son permamentes.
	Estados	• **Activo:** este estado se tiene desde que comienza su ejecucion y se mantiene hasta completa la ultima instrucción o hasta que se produsca un fallo. • **Parcialmente finalizado:** este estado, tambien conocido como parcialmente cometido, se alcanza en el momento posterior a que la transaccion ejecuta su ultima instrucción. • **Finalizado:** este estado, tambien conocido como cometido, se obtiene cuando la transaccion finalizo su ejecucion y sus acciones fueron almacenadas correctamente en memoria secundaria. • **Fallado:** a este estado se arriba cuando la transaccion no puede continuar con su ejecucion normal. • **Abortado:** este estado garantiza que una transaccion fallada no ha producido ningun cambio en la BD, manteniendo la integridad de la informacion alli contenida.
	Metodos de recuperacion de integridad de una base de datos	El problema que se genera ante un fallo es la posible perdida de consistencia en la BD. Esto se debe a que una transaccion puede haberse ejecutado parcialmente sobre la BD. Acciones que se deberian llevar a cabo para asegurar la integridad de la BD: • **Atomicidad:** se debe detectar las instrucciones que es mejor que sean atomicas y definirlas como tal. • **Bitacora:** El metodo de bitacora (log) o registro historico platena que todas las acciones llevadas a cabo sobre la BD deben quedar registradas en un archivo historico de movimientos. La bitacora se puede aplicar en entornos concurrentes. Por otro lado hay que destacar que no siempre se garantiza la consistencia de una BD. • **Bitacora con modificacion diferida de una BD:** Este metodo demora todas las escrituras en disco de las actualizaciones de la BD, hasta que la transaccion alcance el estado de finalizada en bitacora. • **Bitacora con modificacion inmediata de una BD:** Supongan que una transaccion durante su ejecucion

		modifica varios valores de la BD. Los cambios sobre la BD se aplazan hasta que la transaccion finaliza, y en ese momento se deben escribir a disco. Esta accion significa una importante sobrecarga de trabajo cada vez que finaliza la transaccion. Si por el contrario los cambios de la BD, se efecutan a medida que la transacion avanza en su ejecucion, la sobrecarga de trabajo tiene a desaparecer, distribuyendose en el tiempo. La modificacion inmediata de la BD plantea actuar de esta forma. Los cambios se producen a medida que se producen en la transaccion que se esta ejecutan, siempre bajo la consigan que indica que un cambio se registra primero en bitacora para luego grabarse en la BD. Ahora puede suceder que el fallo de una transaccion ocurra cuando ya se produjeron previas modificaciones durante la transaccion. Para solucionar esta inconsistencia se debe manejar otra operación DESHACER cuyo efecto sera retrotraer la BD al estado que tenia antes de comenzar la transaccion. • **Punto de verificacion:** Con el fin de evitar la revision de toda la bitacora ante un fallo, el gesto de la BD agrega en forma periodica, al registro historico, una entrada <punto de verificacion> Estos puntos de verificacion se agregan cuando se tiene la seguridad de la BD esta en estado consistente. La finalidad es indicar que ante un fallo, solo debe revisarse la bitacora desde el punto de verificacion en adelante, dado que las transaccion anteriores finalizaron correctamente sobre la BD. • **Doble paginacion:** El metodo alternativo a bitacora recibe el nombre de doble paginacion o paginacion en la sombra. Este metodo plantea dividir a la BD en nodos virtuales (paginas) que contienen determinados datos. Se generan dos tablas en disco y cada una de las tablas direcciona a los nodos (paginas) generados. Ante una transaccion que realiza una operación de escritura: 1. Se obtiene desde la BD el nodo sobre el cual debe realizarse la escritura (si el nodo esta en memoria principal, este paso se obvia) 2. Se modifica el dato en memoria principal y se graba en disco, en un nuevo nodo, la pagina actual que referencia al nuevo nodo. 3. Si la operación finaliza correctamente, se modifica la referencia de la pagina a la sombra para que apunte al nuevo nodo. 4. Se libera el nodo viejo. En caso de producirse un fallo, luego de la recuperacion, se desecha la pagina actual y se toma como valida la pagina a la sombra.. Ventajas: • Se elimina la sobrecarga producida por las escrituras al registro historico y la recuperacion ante un error es mucha mas rapida. Desventajas:

		• Exije dividir la BD en nodos o paginas y esta tarea puede ser compleja. • Produce sobrecargas en transacciones concurrentes. • Puede generar varios nodos ocupados no referencias. Se desperdicia espacio. Para eso se requiere de algun algoritmo “garbage collector” que detecte estos nodos y los libere.