BASES DE DATOS:

Definición: Conjunto de datos estructurados, relacionados y almacenados en un soporte físico. Su objetivo es el de automatizar:

- El **Mantenimiento**
- Cualquier **informe** de información
- Cualquier **consulta** sobre dicha información

Para crea una base de datos, primero hay que estudiar la información y después organizarla lógicamente.

Historia de las Bases de Datos

Tuvieron sus orígenes en 1960 − 1962, cuando se empezaron a usar las maquinas que codificaban la información en tarjetas perforadas por medio de agujeros. Las bases de datos se crean con el objetivo de almacenar grandes cantidades de datos que antes se almacenaba en libros, lo que era lento, costoso y complejo(cualquier actualización a realizar, había que hacerla en cada uno de los libros en los que apareciera dicha información a modificar).

Las primeras bases de datos manejaban ficheros que eran almacenados en tarjetas o soportes magnéticos. Cuando los ordenadores evolucionan, aparecen las cintas y los discos, a la vez que las maquinas son dotadas de mucha mas potencia y facilidad de manipulación, es por tanto en ese momento cuando las bases de datos comienzan a ser realmente utiles.

En **1970** se convoca una ***Conferencia de Lenguajes de Programación*** y se establece un modelo llamado ***CODASYL*** (Modelo para el tratamiento de bases de datos que fue ***publicado por E. Cod en 1970***. Cod, propuso una forma de organizar las bases de datos mediante un modelo matemático lógico.

Una vez creado este modelo se crea un modelo estandar de actuación.

¿Por qué se crean las bases de datos?

Las bases de datos son compactas, rápidas, cómodas y ofrecen la información actual sometida a la ultima actualización.

Compacta: Lo que hace que una base de datos sea compacta es la centralización de los datos. Es decir que toda la información esta localizada en un solo sitio y a es información se puede acceder desde diferentes sitios, diferentes personas y de diferentes formas.

Rápida: Como la información esta centralizada, cuando queremos actualizar o modificar algún dato, solo deberemos modificarlo una vez.

Cómoda: Al tener la información en un mismo sitio, ahorraremos tiempo y trabajo.

Actual: Debido a la potencia de los ordenadores, estos nos muestran los datos desde la ultima actualización, prácticamente en tiempo real. Reduciendo así los errores

Otras ventajas de las bases de datos:

- Disminuir la Redundancia

- Compartición de Datos
- Posibilidad de aplicar restricciones de seguridad
- Posibilidad de mantener la integridad
- ***Disminuir la Redundancia***

Definimos redundancia como la *duplicación de datos.*

La duplicación de datos genera a su vez una duplicación del trabajo a la hora de mantenerlos y actualizarlos.

Por tanto las Bases de Datos al reducir la duplicación de datos, disminuyen el trabajo. Es fundamental hacer copia de seguridad de la base de datos cada vez que esta quede actualizada.

Si compensa duplicar datos para aumentar la velocidad de la base de datos en cuestión, estaremos en una circunstancia en la que compensará la redundancia de dichos datos.

También puede darse que una duplicación de datos sea obligatoria por las circunstancias. Pero al ser posible *siempre es mejor intentar evitar la redundancia*.

- ***Compartición de Datos***

Hablamos de datos actuales, ya que al ser centralizados, se puede tener acceso a los datos con la ultima actualización en prácticamente tiempo real.

- ***Restricciones de Seguridad***

Para mantener la seguridad a cerca del mantenimiento de los datos, los administradores de la Base de Datos, crean una jerarquía de acceso, que permitirá o prohibirá a los usuarios hacer una u otra acción sobre dicha base de datos.

- ***Integridad***

En una base de datos debemos mantener una coherencia. (No dejar que se introduzcan caracteres en un campo numerico). Esto se contralará mediante:

  - Máscaras
  - Reglas de validación

Ejercicio Práctico

Organizar los datos de entrada para la base de datos de una Biblioteca:

Datos de entrada: Titulo − Dirección − Fecha de Entrega − Codigo Postal − Teléfono − Numero de Páginas − Unidades − Ciudad − Fecha de Admisión − Autor − Categoría − Nombre − Fax − Observaciones − Foto.

Resolución:

Debemos organizar dichos datos de entrada en los grupos que nos interese, mediante un proceso lógico, al tratarse de una Biblioteca, nos debemos centrar como elemento principal en el libro.

Grupo 1: LIBROS

Titulo

Categoría

Autor

Paginas

Unidades

Grupo 2: USUARIOS Grupo 3: PRESTAMOS

Nombre Fecha de Entrega

Dirección Fecha de Devolución

Codigo Postal Observaciones

Fax

Telefono

Foto

Ciudad

Seguna definición de Base de Datos: Colección de datos interrelacionados, almacenados en conjunto sin redundancias perjudiciales. Su finalidad es la de servir a una aplicación o más de la mejor manera posible.

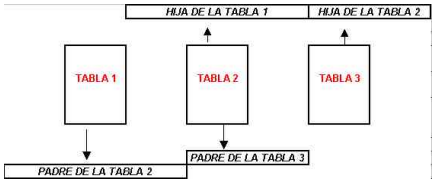
Los datos se almacenan de forma que resulten independientes de los programas que los usan. Se emplean métodos bien determinados para incluir datos nuevos y para modificar o extraer los datos almacenados.

¿ Quien usa las bases de datos?

- Programadores de la aplicación, para manejar la base de datos.
- Usuarios finales: Personas que trabajan mediante el programa aplicación, o lenguajes de consulta en la base de datos.
- Administrador: Se encarga del mantenimiento de la base de datos(Estructura, seguridad e integridad).

¿ Como se relacionan las bases de datos?

Solo puede haber relacion entre una tabla padre y otra tabla hija, de modo que no se puede establecer la relacion directa entre tres tablas, aunque se podrá hacer de la siguiente manera:



Conceptos básicos de bases de datos relacionales

CAMPO 1 CAMPO 2 CAMPO 3 CAMPO 4

Dato	Dato	Dato	Dato
Dato	Dato	Dato	Dato

Una tabla esta estructurada en filas y columnas. Cada fila es un registro de datos, ya que contiene datos. Los datos son la intersección de una fila con una columna y es la unidad mínima con la que podemos trabajar en una base de datos. El gestor Acces colocará los datos en una matriz donde las filas serán los registros y las columnas los campos.

- Base de Datos: Conjunto de tablas relacionadas entre si.
- Tabla: Objeto que almacena datos en registros y campos.
- 2ª Definición de Tabla: Una tabla es un objeto que almacena la información referida a un tema en concreto dentro del conjunto de temas que forman un modelo de datos.
- Modelo de datos: El conjunto de tablas o temas y las relaciones entre ellas, que se han de almacenar en la base de datos.
- Objeto: Conjunto de cosas que se tratan como una unidad.
- Entidad: Agrupación lógica (Otra forma de referirse a Tabla)
- Tema: Otro termino que significa lo mismo que Tabla
- Campo: Un campo almacena un tipo de información o categoría, es decir, un componente de una tabla que contiene un elemento específico de información.
- Atributo: Determinada información sobre un tema específico.
- Registro: Un registro almacena toda la información relativa a un elemento o sujeto de la base de datos.Es un conjunto de datos acerca de un sujeto de la base de datos.
- Datos: Intersección de un registro y un campo. Unidad mínima de información que almacenamos en una base de datos.

Clave Principal: Campo de la tabla que identifica inequívocamente un registro, no pudiendo existir dos registros con la misma clave principal. Esta clave sirve para identificar un registro en concreto. También llamada Primary Key.

El campo clave no puede tener un valor nulo (NULL). Su valor debe ser conocido. El contenido del campo no deber ser extenso, el espacio que ocupe deberá ser mínimo. Su utilización en otras tablas (Claves Ajenas) sirve para crear una relación entre ellas.

Clave Ajena: Duplicación en la tabla relacionada de la clave principal. Si el campo definido como clave principal o primaria en una tabla forma parte también de otra tabla se llamará clave ajena. También llamada Foreign Key.

- Es muy conveniente que cada tabla contenga una clave principal.
- Relación: Es una asociación establecida entre dos tablas gracias a dos campos comunes a estas. La relación se establece entre la clave principal de una tabla (Tabla principal) y la clave ajena de otra tabla que sera la tabla relacionada.

Tipos de Relaciones

1 : 1 ..... Relación de 1 registro a 1 registro.

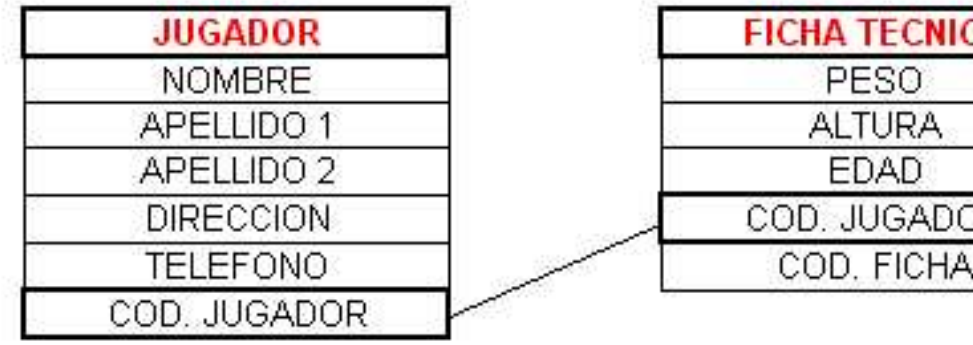
1 : n ..... Relación de 1 registro a varios registros. Otras representaciones:

- 1 a Muchos
- 1 a Varios
- 1 : "
- 1 : M

n : n ..... Relacion de varios registros a varios registros. Otras representaciones:

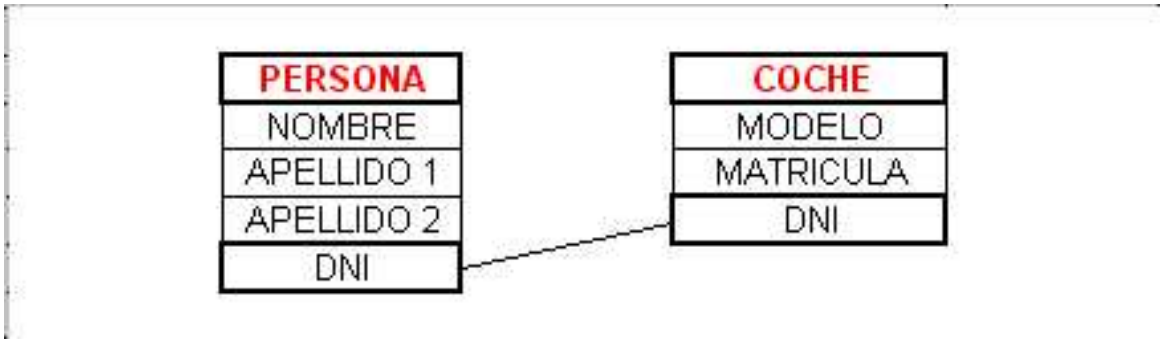
- Varios a Varios
- " : "
- M : M

- Dos cifras separadas por dos puntos, la primera cifra nos indicará el numero de registros de la primera tabla que se vana relacionar y la segunda nos indica el número de registros de la segunda tabla con los que se pueden relacionar.
- 1: 1 : Relación de tal manera que, un registro de la tabla principal solo se va ha relacionar con un registro de la tabla relacionada y a la inversa.
  Ej.:



Con esta relación a cada jugador siempre le va ha corresponder una única ficha, y cada ficha siempre le corresponderá un único jugador. Este tipo de relación no suele usarse debido a que si juntáramos las dos tablas no crearíamos redundancia perjudicial y además ahorraríamos tiempo en las consultas.

- 1 : n : Esta relación se da cuando a un único registro le corresponden varios registros. Ej.:



Con esta relación, a una misma persona le podrían corresponder mas de un coche. De modo que a cada coche le corresponderá mas de un registro de la tabla Persona.

- n : n : No se puede implementar en Access, ni se debe hacer. Esta relación se basa en que varios registros de la tabla principal pueden estar relacionados con varios registros de la tabla relacionada.

TIPOS DE DATOS PARA LOS CAMPOS DE UNA TABLA

- ***TEXTO***: Admite contenido de caracteres alfanuméricos, el tamaño estará entre 1 y 255 Bytes y el valor por defecto de esta tamaño es de 50 Bytes.

- ***MEMO***: Admite contenido de tipo alfanumérico, el tamaño esta entre 1 y 64.000 Bytes (64Kas).

- ***NUMERICOS***: Admite contenido de caracteres numéricos, el tamaño será entre 1, 2, 4 y 8 Bytes, dependiendo del formato de tipo numérico. El formato de tipo numérico puede ser de tipo: DOUBLE, FLOAT, INTEGER, etc... Este tipo de datos solo se utilizara cuando vayamos a realizar operaciones con ellos.

- ***FECHA Y HORA***: Solo admite horas y fechas. Este tipo de dato ocupa 8 Bytes.

- ***MONEDA***: Admite contenido de caracteres numéricos, dándoles a estos un formato automático para una moneda (Puntos de millar, símbolo de la peseta, etc..) dependiendo este formato de la configuración regional del panel de control. Este tipo de datos puede llevar decimales y se utilizara siempre que hablemos de cantidades monetarias.

- ***AUTONUMERICO***: Este campo lo crea Access con nuestro consentimiento de una manera automática, para establecerlo como una clave principal. Es de tipo numérico, y es incrementada automáticamente por Access cada vez que añadimos un nuevo registro a la tabla.

Este tipo de claves principal, no suelen usarse, ya que es preferible definir nuestras propias claves, aunque puede ser usada como una posible puerta trasera en alguna ocasión.

- ***SI / NO***: Tipo de datos boléanos , los valores boléanos pueden tener dos valores posibles, 1 o 0, donde 1 es verdadero y 0 es Falso.

- ***OBJETO OLE***: El contenido serán gráficos y el tamaño de estos puede ser de hasta 1 un Gygabyte. Hay que tener especial cuidado con estos objetos, pues al borrarlos de la base de datos, no se reduce el tamaño de la misma, mientras esta no sea compactada.

Si damos de baja un registro en una base de datos Access, el programa debe compactar la base de datos para reducir y eliminar los espacios en blancos que este registro a producido al ser eliminado. De este modo se reduce el tamaño de la base de datos. Lo mismo pasa con los objetos OLE.

INTEGRIDAD REFERENCIAL

Podemos definir integridad referencial al hecho de no hacer referencia a registros o campos que no existen por que nunca fueron creados o por que han dejado de existir en el transcurso de la base de datos.

Por tanto podemos afirmar que la integridad referencial impide que queden registros huérfanos, ya que no podemos hablar de una persona que figura en la tabla de ventas si esta no existe en la tabla de clientes.

La integridad referencial garantiza que los cambios efectuados en una tabla no afectaran a la relación.
Además impide:

- Añadir un registro a la tabla seleccionada si el valor del campo de relación no existe en la tabla principal.
- Eliminar un registro de la tabla principal si tiene registros relacionados en la tabla relacionada. Esto quiere decir, siguiendo con el ejemplo anterior, que antes de borrar a una persona de la tabla de clientes, habría que borrarla de la tabla de ventas, ya que no pueden existir ventas de un cliente inexistente.
- Modificar el campo de relación en la tabla principal si tiene registros relacionados en la tabla

relacionada.

- ♦ Modificar el campo de relación en la tabla relacionada si el nuevo valor que se indexa en el no existe en la tabla principal, ya que este debe estar también en la tabla principal.

Si exigimos ***integridad referencial se van a activar las siguientes opciones***:

- Actualizar en cascada los campos relacionados:

Al activar esta opción, si realizamos alguna modificación en el campo clave principal de la tabla principal, Acces realizara una actualización automática en el campo relacionado de la tabla relacionada.

- Eliminar en cascada los registros relacionados:

Si eliminamos un registro en la tabla principal, se eliminaran automáticamente todos los registros relacionados en las demás tablas relacionadas.

PROPIEDADES DE LOS CAMPOS EN VISTA DE DISEÑO

Son las siguientes:

Tamaño del Campo: Se puede especificar para campos cuyo tipo de datos sea texto o numérico. Esta especificado en Bytes. Para el resto de los campos ya tienen un valor por defecto que varia según los mismos:

- ♦ ***Campos de texto***: Indica el número máximo de caracteres que se pueden escribir, ya que cada carácter ocupa 8 bits en memoria, lo que equivale a un Byte. El valor por defecto es 50. Hay que intentar ajustar el tamaño al máximo ya que cuando la base de datos va creciendo, la mala asignación de los mismos puede provocar falta de espacio para almacenar la información.

Propiedad permitir longitud 0: Es booleana. Si la establecemos en verdadero , va ha permitir al usuario introducir una cadena vacia para un campo de texto.

- ♦ ***Campos Numéricos***: Rango de números que se podrán introducir, además de el numero de decimales. Para los campos numéricos existen unos formatos específicos:

- *BYTE*: Tiene un rango que va desde el 0 al 255
- *ENTERO*: Tiene un rango desde −32768 a 32767
- *ENTERO LARGO*: Tiene un rango de 2147483 a 2147483
- *DECIMAL SIMPLE*: Rango desde −3,4*10e38 a 3,4*10e38
- *DECIMAL DOBLE*: Rango desde −10e308 a 10e308

Propiedad de Formato: Determina el aspecto en que serán presentados los datos. Solo afecta a la presentación y no a los datos almacenados en la tabla. Si no aplicamos ningún formato de presentación los datos de representaran tal y como nosotros los hemos almacenado.

Existen unos *formatos predefinidos*, son los siguientes:

- Formatos de tipo **FECHA / HORA** :

- ♦ FECHA GENERAL: 21/10/99 10:36:00
- ♦ FECHA LARGA: domingo, 19 de Junio de 1999
- ♦ FECHA MEDIANA: 19−jun−99
- ♦ FECHA CORTA: 19/06/99

- ♦ HORA LARGA: 10:36:00 p.m
- ♦ HORA CORTA: 10:36
- ♦ HORA MEDIANA: 10:36 p.m

- Formatos de tipo **SI / NO**:

  - ♦ Su formato puede ser de tres tipos:
  - ♦ SI / NO
  - ♦ VERDADERO / FALSO
  - ♦ ACTIVADO / DESACTIVADO
  - ♦ PERSONALIZADO:

  Podemos crear un formato personalizado indicando dos valores separados por comas, donde el primero corresponderá a Verdadero y el segundo a Falso. Ej.:

  Pepe ; Juana

  − Formatos de tipo **MONEDA:**

  Los formatos predefinidos son:

  · Número General: En este formato los datos se representan tal y como se han almacenado.
  · Moneda: Presenta los datos almacenados incluyendo punto de millares, dos decimales y el sufijo PTS.
  · Fijo: Presenta al menos un digito y tendrá dos posiciones decimales.
  · Estándar: Incluye el punto de millares y dos posiciones decimales.
  · Porcentaje: Multiplica por 100, presenta dos decimales y añade eñ símbolo %.
  · Científico: Utiliza los números según la notación exponencial.

  − Formatos de tipo **TEXTO y MEMO**:

  No hay formatos predefinidos.

  Propiedad de lugares decimales: Numero de decimales de 0 a 15, aparecerán tantas cifras como se indique, sin tener en cuenta las que se especifican en el formato. Tiene como valor predeterminado *AUTOMATICO,* este valor usa los decimales predeterminados de cada formato.

  Propiedad de titulo: Se utiliza la propiedad titulo para establecer una etiqueta con un máximo de 255 caracteres que aparecerá en los formularios o informes en los cuales se analiza el campo para el que se ha definido dicho titulo. También aparecerá este título en la ventana de hoja de datos.

  Propiedad Valor predeterminado: Esta propiedad sirve para indicar que valor por defecto debe añadir Acces para ese campo al agregar un registro. El valor predeterminado sirve para ahorrar tiempo a los usuarios poniendo un valor normal como default.

  Propiedad de regla de validación: Establece la condición o condiciones que han de ampliar los datos que se introducen en los campos, evitando así posibles errores. De este modo se podrían evitar números negativos en un campo que contenga edades.

Estas reglas se establecerán mediante expresiones que no vamos a ver en este momento, y que descubriremos mas adelante.

Propiedad de texto de validación: Es el mensaje que aparecerá en pantalla si el dato introducido para un campo no cumple la regla de validación. Se establece escribiendo el mensaje en la caja adecuada. Si no escribimos nada, Acces mostrara uno predeterminado.

Propiedad de Requerido: Si se especifica SI en esta propiedad, Acces obliga al usuario a introducir un valor en este campo para cada registro, es decir, no permite valores nulos. Cualquier campo excepto la clave principal, tiene como valor predeterminado NO.

Propiedad de la mascara de entrada: Patrón que se utiliza cuando los datos que se van a introducir tienen valores similares, estas mascaras facilitan la introducción de datos y previenen errores al impedir la entrada de datos no validos. Ej.: Impedir que se introduzcan letras en un campo de telefono.

Para establecer las mascas se utilizan una serie de caracteres especiales:

0 : Representa un numero de introducción obligatoria.

9: Representa un numero de introducción no obligatoria.

L: Representa una letra de introducción obligatoria.

?: Representa una letra de introducción no obligatoria.

A: Representa una letra o numero de introducción obligatoria.

a: Representa una letra o un numero de introducción no obligatoria.

&: Cualquier carácter o espacio de introducción obligatoria.

C: Cualquier carácter o espacio de introducción no obligatoria.

<: Convierte los caracteres a minúsculas.

>: Convierte los caracteres a mayúsculas.

LAS EXPRESIONES

Una expresión se forma de distintos componentes:

- Operadores
- Literales
- Identificadores
- Funciones

OPERADORES:

Aritméticos: Se utilizan para operaciones matemáticas básicas.

\ : Este operador efectúa una división de enteros.

MOD: Devuelve el resto de la división.

Lógicos:

XOR: Para ser verdadero los valores tienen que ser diferentes.

AND: verdaderos.

NOT: Invierte el símbolo 1 a 0 y 0 a 1.

OR: Cualquiera de los símbolos debe ser verdadero.

A	**B**	**AND**	**OR**	**XOR**	**NOT A**	**NOT B**
0	***0***	0	0	0	1	1
0	***1***	0	1	1	1	0
1	***0***	0	1	1	0	1
1	***1***	1	1	0	0	0

Concatenación:

Pepe & Perez

&: Este símbolo concatena cadenas.

+: Igual que &, pero NO debe usarse para evitar confusiones con valores matemáticos.

Identificación:

Estos operadores nos permiten trabajar a nivel de objeto:

Ej.:

Clientes!Nombre.InputMask = ...

Donde Clientes seria la clase de objeto(tabla, campo..) en este caso una tabla.

Nombre: El nombre del objeto.

InputMask: Mascara de entrada.

Otros operadores:

Es − Is : Podemos usar este operador para comprobar NULL, EMPTY, TRUE, FALSE.

Como − like : Comparar con una serie de patrones o comodines(*,?)

Entre − Between: Comprobar si estamos dentro de un rango determinado.

Ej.:

BETWEEN #01/01/00# AND #31/12/00#

Se ponen almohadillas para indicar a Acces que se trata de una fecha.

La fecha debe estar dentro del año 2000.

En − In: Permite comprobar si estamos dentro de unos posibles valores. Ya sea una lista o un rango. Ej.:

IN (A,B,C);

VALORES LITERALES:

Numéricos: Se expresa con dígitos cuando sea necesario el separador decimal y cuando sea necesario el signo. Ej.:

1234

1234

−1234

−123,4

Texto: Se encierran entre comillas dobles Hola. Para representar el Intro recurrimos al valor 13 (código ASCII). Mediante el comando:

Car$(13)

Print Hola & Car$(13) & Pepe

Car$(10)

Fecha: Un dato de tipo fecha y hora se encierra entre almohadillas.

Ej.:

#14/01/00#

= Date () ; Nos da la fecha actual

12