Tema 2. Bases de Datos Relacionales

Bases de Datos

1º DAW – Curso 25/26

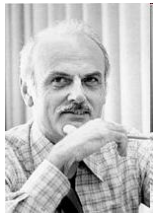
Profesor Juan Carlos Moreno

IES Ntra. Sra. Los Remedios

1. Modelo Relacional

- Una Base de Datos Relacional es una base de datos que se construye en base a las especificaciones del Modelo Relacional.
- El Modelo Relacional es el modelo más utilizado para representar sistemas dinámicos de información, se impuso dada las limitaciones de los modelos anteriores: modelo en red y jerárquico, pero sobre todo por su sencillez y robustez.
- Se basa:
  - Uso de las tablas para almacenar los datos
  - Conexiones entre las tablas para relacionar los datos
  - Conjunto de restricciones (o condiciones) que deberán cumplir dichos datos

1.1 Origen.



- En 1970 Edgar Frank Codd publicó un artículo en Comunications of the ACM, proponiendo un nuevo modelo de datos que tenía como objetivo fundamental aislar al usuario de las estructuras físicas de los datos, consiguiendo así la independencia de las aplicaciones respecto de los datos.
- El nuevo modelo se basa en la teoría matemática de las relaciones. Los datos se estructuran lógicamente en forma de relaciones (muy parecido al concepto de tabla).
- Aunque trabajaba para IBM, esta empresa no recibió de buen grado sus teorías (de hecho continuó trabajando en su modelo en red IMS).
- De hecho fueron otras empresas (en especial Oracle) las que implementaron sus teorías.
- Pocos años después el modelo se empezó a utilizar cada vez más, hasta finalmente ser el modelo de bases de datos más popular. Hoy en día casi todas las bases de datos siguen este modelo.

1.2 Objetivos del Modelo

- **Independencia física:**
  - El modo en que se almacenan los datos no influye en su manipulación lógica. No será necesario modificar las aplicaciones por cambios en el almacenamiento físico
- **Independencia lógica:**
  - Cualquier modificación en los distintos objetos de la base de datos, no repercuta ni en los usuarios finales ni en las aplicaciones que están accediendo a la bbdd.
- **Flexibilidad**.
  - La base de datos ofrece fácilmente distintas vistas en función de los usuarios y aplicaciones.
- **Uniformidad y Sencillez**:
  - Tablas: los usuarios ven la base de datos relacional como una colección de tablas, y al ser la tabla la estructura fundamental del modelo, éste goza de una gran uniformidad
  - SQL: lo que unido a unos lenguajes no navegacionales y muy orientados al usuario final, da como resultado la sencillez de manejo de los sistemas relacionales.
- **Fundamentación teórica**:
  - Al estar el modelo definido con rigor matemático, ya que está basado en la teoría matemática de las relaciones.

1.3 Situación y Evolución

- Las características del modelo relacional han hecho que prácticamente todos los SGBD comerciales y libres implementen el modelo relacional:
  - Algunas de las principales empresas informáticas del mundo son, en origen, empresas de SGBD-R: ORACLE, Sybase, INFORMIX, postgresql, Access, MySQL
- El tremendo éxito del modelo relacional ha supuesto que el cambio tecnológico a la siguiente generación esté siendo evolutivo y no revolucionario:
  - Triunfan los *SGBD Objeto-Relacionales*, y
  - Fracasan, en general, los SGBD de Objetos puros.

2. Elementos Modelo Relacional

- ***Relación***: es la estructura básica del modelo relacional. Se representan utilizando *tablas*.
- ***Atributo***: son las propiedades de la relación. Se representan mediante *columnas* en las tablas.
- ***Dominio***: conjunto de valores sobre los que se define el *tipo* de un atributo.
- ***Tupla***: ocurrencia de la relación. Se representa mediante *filas* dentro de las tablas.
- **Clave Primaria**: atributo que identifica de forma unívoca cada tupla de la relación.

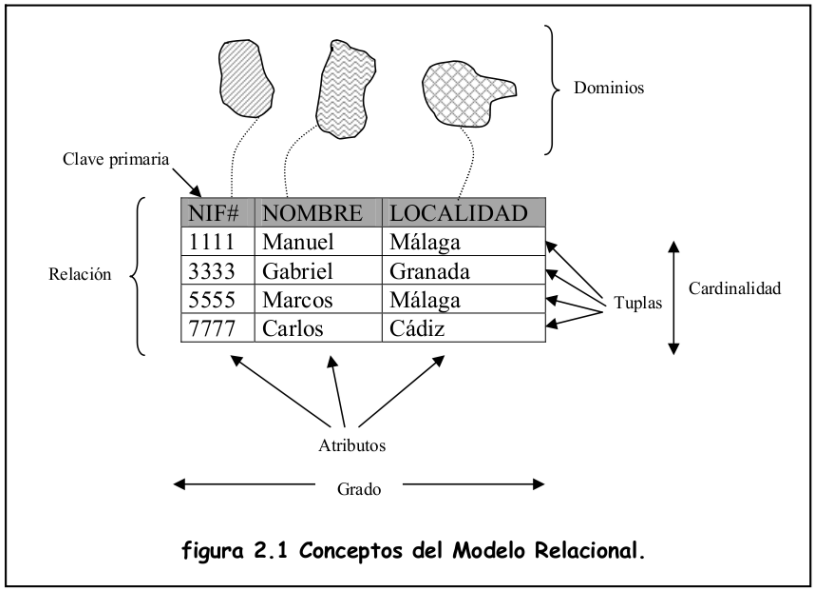


figura 2.1 Conceptos del Modelo Relacional.

2.1 Relación *(Tabla)*

- La **Relación** es el elemento fundamental del Modelo Relacional y está basado en el concepto de tabla, conjunto de filas y columnas.
- Estructura en la que se basa el **Modelo Relacional** para almacenar la información sobre los distintos objetos que se representan en la base de datos.
- No hay que confundir la idea de relación según el modelo de **Codd**, con lo que significa una relación en el modelo Entidad/Relación de **Chen.**
- Puesto que una relación se representa como una tabla; podemos entender que las columnas de la tabla son los atributos; y las filas, las tuplas. Conceptos que veremos más adelante.

atributo 1	atributo 2	atributo 3	....	atributo n	
valor 1,1	valor 1,2	valor 1,3	....	valor 1,n	← tupla 1
valor 2,1	valor 2,2	valor 2,3	....	valor 2,n	← tupla 2
.....	.....	.....	....	.....	....
valor m,1	valor m,2	valor m,3	....	valor m,n	← tupla m

2.1.1 Características Relación

- Vermos a continuación los conceptos de:
  - Grado
  - Cardinalidad
- El ***grado*** de una relación (tabla) es el número de **atributos** que posee. Tipos de relación:
  - Binaria
  - Ternaria
  - N-aria
- La ***cardinalidad*** de una relación (tabla) es el número de **tuplas** que posee

2.1.2 Relación Vs Tablas

- Relaciones vs. Tablas

Relación (modelo teórico)	Tabla (implementación)
Tupla	Fila
Atributo	Columna
Grado	Nº de columnas
Cardinalidad	Nº de filas

Relación ≠ Tabla

2.2 Tupla

- Cada una de las filas de la relación. Se corresponde con la idea clásica de **registro**.
- Representa por tanto cada elemento individual de esa relación.
- Tiene que cumplir que:
  - Cada tupla se debe corresponder con un elemento del mundo real.
  - No puede haber dos tuplas iguales (con todos los valores iguales).

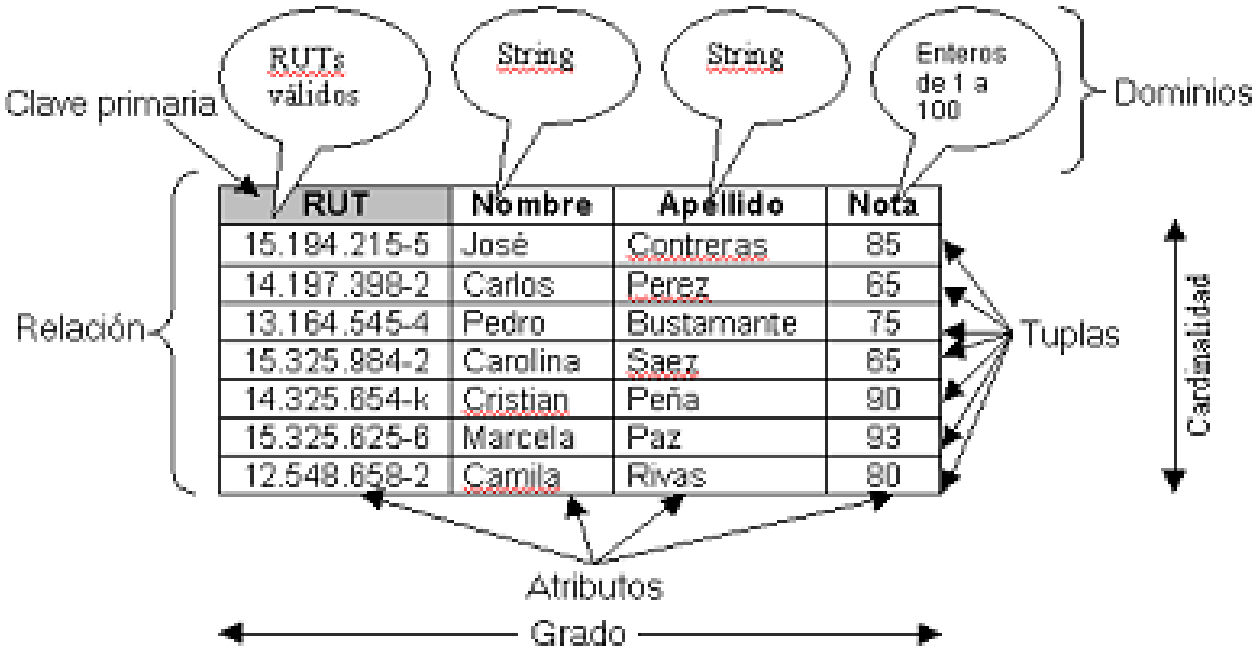
2.3 Atributo *(columna)*

- Un atributo representa una propiedad de una relación.
- Un atributo tomará valores dentro de un domino.
- Distintos atributos de una relación, e incluso de distintas relaciones, pueden tomar valores dentro de un mismo dominio.
- Características atributos o columnas:
  - El número de columnas por tabla es fijo
  - Cada columna ha de ser única y con un nombre único en esa tabla
  - No puede haber columnas duplicadas
  - El valor de cada columna ha de estar contenido en el dominio que representa, aunque puede ser un valor NULO.

2.4 Dominio

- Un dominio contiene todos los posibles valores homogéneos (del mismo tipo) y atómicos (indivisibles) que puede tomar un determinado atributo.
- Los valores contenidos en una columna pertenecen a un dominio previamente definido.
- Dos atributos distintos pueden tener el mismo dominio.
- Todos los dominios tienen un nombre y un tipo de datos asociado.
- Tipos de dominio:
  - **Generales**.
    - Los valores están comprendidos entre un máximo y un mínimo, por ejemplo precio de un producto valor comprendido entre 0 y valor máximo permitido para precio.
  - **Restringidos**.
    - Sólo pueden tomar un conjunto de valores específicos, por ejemplo Sexo que podrá tomar los valores H o M.

Elementos Relación



BBDD Relacional

SUMINISTRADOR

SNUM	SNOM	TIPO	MUN
S1	PÉREZ	30	CERRO
S2	RAMOS	10	PLAYA
S3	ARENAS	20	CERRO
S4	VALLE	20	PLAZA
S5	LÓPEZ	15	PLAZA

PRODUCTO

PNUM	PNOM	PRECIO	PESO
P1	CLAVO	0.10	12
P2	TUERCA	0.15	17
P3	MARTILO	3.50	80
P4	TORNILLO	0.20	10
P5	ALICATE	2.00	50
P6	SERRUCHO	4.00	90

SP

S	P	CANT
S1	P1	3
S1	P2	2
S1	P3	4
S1	P4	2
S1	P5	1
S1	P6	1
S2	P1	3
S2	P2	4
S3	P3	4
S3	P5	2
S4	P2	2
S4	P4	3
S4	P5	4

2.5 Definición Formal

- Una relación se componen del Esquema (intensión) y de la extensión.
  - Esquema corresponde a la cabecera de la tabla
  - Extensión corresponde al cuerpo
- **Esquema**:
  - Consiste en un nombre de relación *R* y un conjunto de atributos *(A_1, A_2, …, A_n). Ejemplo Relación Empleados: Empleados(DNI, Nombre, Apellidos, Sueldo)*
- ***Extensión*** del esquema *$R(A_1, A_2, …, A_n)$* es un conjunto de tuplas *t_i (i = 1, 2, ..., m)* donde cada tupla *t_i es a su vez un conjunto de pares t_i = {<A_1:v_{i1}>, <A_2:v_{i2}> ... >A_n:v_{in}>} y, para cada par <Aj:vij>, se cumple que v_{ij} es un valor de dominio(A_j), o bien un valor especial que denominaremos nulo.*

Ejemplos

Empleados			
DNI	*nombre*	*apellido*	*sueldo*
40.444.255	Juan	García	2.000
33.567.711	Marta	Roca	2.500
55.898.425	Carlos	Buendía	1.500

▶Esquema

▶Extensión

Clientes		
DNI	**Nombre**	**Edad**
12333944C	Ana	52
12374678G	Eva	27
28238232H	Martín	33

Esquema: **Cliente**(DNI:**DNI**, Nombre:**Nombre**, Edad:**Edad**)

Cuerpo: {(DNI: "12333944C", Nombre:"Ana", Edad:52), (DNI: "12374678G", Nombre:"Eva", Edad;52), (DNI: "28238232H", Nombre:"Martín",Edad:33)}

3 Claves

- En una relación no hay tuplas repetidas. Se identifican de un modo único mediante los valores de sus atributos. Toda fila debe estar asociada a una clave que permita identificarla. A veces la fila se puede identificar por un único atributo, pero otras veces es necesario recurrir a más de un atributo.
- La clave debe cumprir los siguientes requisitos:
  - **Identificación unívoca**: En cada fila de la tabla el valor de la clave ha de identificarla de forma unívoca o inequívoca, ha de ser único.
  - **No redundancia**. No se puede descartar ningún atributo de la clave para identificar la fila.
- Tipos:
  - Claves Candidatas
  - Clave Primaria
  - Clave Alternativa o secundaria
  - Clave externa o ajena

3.1 Claves Candidatas

- Atributo o conjunto de atributos que identifican unívoca y mínimamente cada tupla de la relación.
- Toda tabla en el modelo relacional debe tener al menos una clave candidata, ya que por definición del modelo no puede haber dos tuplas iguales.
- Puede ocurrir que una relación tenga más de una clave candidata
- En el ejemplo de la figura:
  - Claves Candidatas: Nempleados, Dni

Empleados : Tabla

		NEmplead	Nombre	Apellidos	Dni	Domicilio	Poblacion	CP	FechaIngreso	Departame	CodCateg	CargoPer	Casado	Observ
▶	+	1	GEMA	DIAZ DIAZ	10.111.222-A	LA SOTA 2	BARREDOS	33970	11/12/2001	TEC	1	3	☑	
	+	2	ENRIQUE	BORBON CASTRO	10.222.333-B	RIO EO 1	POLA DE LAVIANA	33980	10/11/2000	OBD	1	1	☐	
	+	3	PABLO	SANTOS GINER	10.333.444-C	ASTURIAS 2	GIJÓN	33211	31/12/1998	PRO	1	3	☑	
	+	4	ELENA	DOSAL GOMEZ	10.444.555-F	URIA 4	GIJON	33213	25/07/1990	ANA	2	1	☑	
	+	5	SANTIAGO	SEGURA TORRENTE	10.555.666-G	CONDE SIZZO	LA FELGUERA	33960	31/07/1967	ANA	3	2	☐	
	+	6	MARTA	MIRALLES KENNER	10.666.777-H	AVDA. PRINCIPAD	OVIEDO	33011	13/08/1995	OBD	4	2	☐	
	+	7	ROCIO	FERRER GOMEZ	10.777.888-J	LAS XANAS, 2	GIJON	33212	14/07/2000	TEC	5	1	☐	
✳		0					Ubrique				0	0	☐	

3.2 Clave Primaria o Principal

- Aquella clave candidata que el usuario escoge para identificar las tuplas de la relación.
- Si sólo existe una clave candidata, ésta se eligirá como Clave Primaria.
- En caso de tener varias claves candidatas, ¿Cuál se elige como Clave Principal?:
  - La candidata que identifique mejor a cada tupla en el contexto de la base de datos.
  - Ha de ser lo más mínima posible, en el sentido de que en su composición no intervengan más que los atributos estrictamente requeridos para identificar las tuplas de forma única.
  - Normalmente se usan Claves Numéricas, que ocupan menos espacio.
- En el ejemplo de la figura:
  - Claves Candidatas: NEmpleados, Dni
  - Clave Principal: NEmpleados

Empleados : Tabla

		NEmpleado	Nombre	Apellidos	Dni	Domicilio	Poblacion	CP	FechaIngreso	Departame	CodCateg	CargoPer	Casado	Observ
▶	+	1	GEMA	DIAZ DIAZ	10.111.222-A	LA SOTA 2	BARREDOS	33970	11/12/2001	TEC	1	3	☑	
	+	2	ENRIQUE	BORBON CASTRO	10.222.333-B	RIO EO 1	POLA DE LAVIANA	33980	10/11/2000	OBD	1	1	☐	
	+	3	PABLO	SANTOS GINER	10.333.444-C	ASTURIAS 2	GIJÓN	33211	31/12/1998	PRO	1	3	☑	
	+	4	ELENA	DOSAL GOMEZ	10.444.555-F	URIA 4	GIJON	33213	25/07/1990	ANA	2	1	☑	
	+	5	SANTIAGO	SEGURA TORRENTE	10.555.666-G	CONDE SIZZO	LA FELGUERA	33960	31/07/1967	ANA	3	2	☐	
	+	6	MARTA	MIRALLES KENNER	10.666.777-H	AVDA. PRINCIPAD	OVIEDO	33011	13/08/1995	OBD	4	2	☐	
	+	7	ROCIO	FERRER GOMEZ	10.777.888-J	LAS XANAS, 2	GIJON	33212	14/07/2000	TEC	5	1	☐	
✳		0					Ubrique				0	0	☐	

3.3 Claves Alternativas o Secundarias

- Cualquier clave candidata que no ha sido escogida como Clave Prinaria
- En nuestro ejemplo:
  - Claves Candidatas: Nempleados, Dni
  - Clave Principal: Nempleados
  - Clave Alternativa o Secundaria: Dni

Empleados : Tabla

		NEmplead	Nombre	Apellidos	Dni	Domicilio	Poblacion	CP	FechaIngreso	Departame	CodCateg	CargoPer	Casado	Observ
▶	+	1	GEMA	DIAZ DIAZ	10.111.222-A	LA SOTA 2	BARREDOS	33970	11/12/2001	TEC	1	3	☑	
	+	2	ENRIQUE	BORBON CASTRO	10.222.333-B	RIO EO 1	POLA DE LAVIANA	33980	10/11/2000	OBD	1	1	☐	
	+	3	PABLO	SANTOS GINER	10.333.444-C	ASTURIAS 2	GIJÓN	33211	31/12/1998	PRO	1	3	☑	
	+	4	ELENA	DOSAL GOMEZ	10.444.555-F	URIA 4	GIJON	33213	25/07/1990	ANA	2	1	☑	
	+	5	SANTIAGO	SEGURA TORRENTE	10.555.666-G	CONDE SIZZO	LA FELGUERA	33960	31/07/1967	ANA	3	2	☐	
	+	6	MARTA	MIRALLES KENNER	10.666.777-H	AVDA. PRINCIPAD	OVIEDO	33011	13/08/1995	OBD	4	2	☐	
	+	7	ROCIO	FERRER GOMEZ	10.777.888-J	LAS XANAS, 2	GIJON	33212	14/07/2000	TEC	5	1	☐	
*		0					Ubrique				0	0	☐	

3.4 Clave Ajena *(foránea o Foreign Key)*

- Se denomina clave ajena de una relación R_1 al atributo o conjunto de atributos cuyos valores han de coincidir con los valores de la Clave Principal de la tabla R_2. Ambas claves están definidas sobre el mismo dominio y son muy importantes en el estudio de la integridad de los datos en el modelo relacional.
- Las Claves Ajenas dejan constancia de que existe una conexión o relación entre las tablas R_1 y R_2. La tabla principal es la que cede su Clave Primaria R_2 y la que la recibe en forma de clave ajena, es la tabla secundaria R_1.
- Características de las Claves Ajenas:
  - El dominio ha de coincidir con el dominio de la clave primaria correspondiente
  - Se permite que el nombre sea distinto al de la clave primaria correspondiente
  - Su valor puede estar duplicado o ser nulo
  - Los campos que la forman pueden formar parte de la clave principal de la tabla secundaria

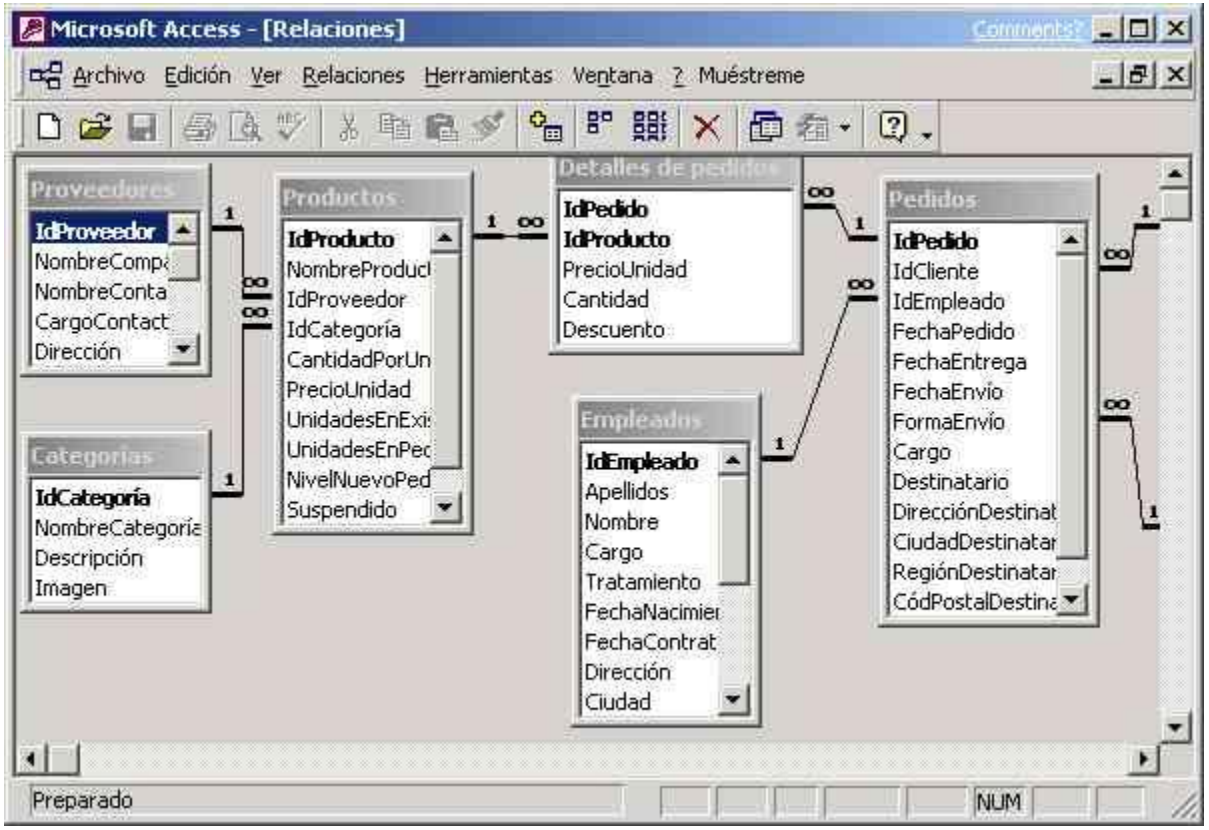
Ejemplo Clave Ajena

- En nuestro ejemplo Nº Equipo de la tabla Equipos sería Clave Ajena. Los valores de dicha clave ajena han de coincidir con los valores de la Clave Principal (Nº Equipo) de la Tabla Equipos.
- Existe una relación entre Equipos y Jugadores, de forma que un jugador ha de pertenecer a un equipo. Tabla Principal es Equipo mientras la Tabla Secundaria es Jugadores.

Equipo	Nº Equipo
Real Madrid	1
F.C. Barcelona	2
Athletic Bilbao	3

Nº Jugador	Jugador	Nº Equipo
1	Karanka	3
2	Ronaldinho	2
3	Raul	1
4	Beckham	1

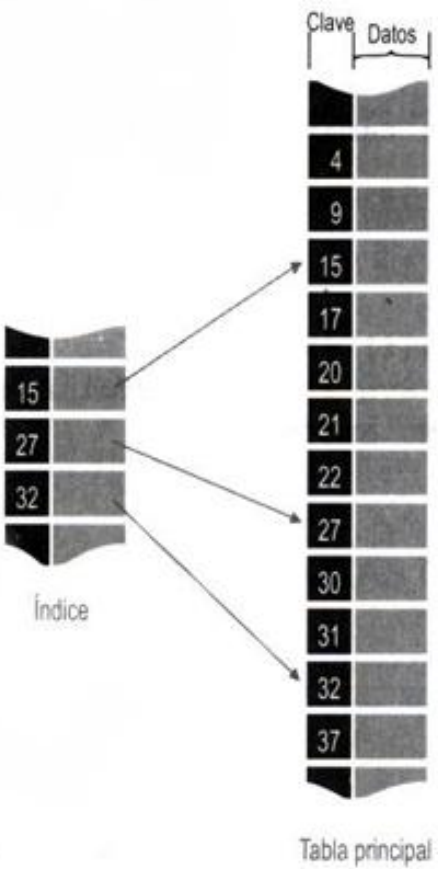
Esquema de una Base de Datos Relacional en Access.



4 Índices

- Es un componente fundamental de las Bases de Datos Relacionales ya que permite mayor velocidad de acceso a los datos.
- Los ficheros índices contienen dos atributos:
  - Campo Índice:
  - Campo Posición: La posición que ocupa en la Tabla el Valor Dado del Campo Índice.
- Toda relación tiene al menos un fichero índice, y es el correspondiente a la Clave Primaria.
  - Campo índice: Clave Primaria
  - Campo Posición Posición que ocupa en la tabla el valor dado de la Clave Primaria.
- Presentan grandes ventajas:
  - Cuando hay que efectuar recorridos por grande tablas ya que se recorren los índices en vez de la tabla original
  - En consultas y actualizaciones, donde primero se ordenan los datos y segundo se seleccionan o actualizan
  - En las búsquedas y selección.
- Sobre una misma tabla se podrán crear tantos índices como necesitemos, aunque tampoco conviene sobrecargar mucho la bas de datos.

Create Index CliApellidos ON Clientes(Apellidos)



5 Valores Nulos

- En los lenguajes de programación se utiliza el valor nulo para reflejar que un identificador (una variable, un objeto,..) no tiene ningún contenido.
- Las bases de datos relacionales significa:
  - En un atributo indica que la tupla en cuestión carece de valor en dicho atributo
  - En claves ajenas indican que el registro actual no está relacionado con ninguno.
- Es importante indicar que el texto vacío **‘’**, no significa lo mismo que un valor nulo; como tampoco el valor cero significa nulo.
- Eso significa definir un tercer valor en la lógica booleana, además de los clásicos **verdadero** y **falso**. Un valor nulo no es ni verdadero ni falso sino ***quizás***.

6. Restricciones del Modelo Relacional

- En todos los modelos de datos existen restricciones que a la hora de diseñar la BBDD han de tenerse en cuenta.
- Los datos almacenados en la BBDD han de adaptarse a la estructura impuesta por el modelo y deben cumplir una serie de reglas o condiciones para garantizar la consistencia y validez de los datos.
- Algunas de ellas ya han sido citadas en las pripiedades de las relaciones y las claves. Existen dos tipos de restricciones:
  - Restricciones Inherentes al Modelo
  - Restricciones Semánticas o de Usuario

Ejemplo

- Errores detectados:
  - Hay dos empleados con el mismo dni.
  - Hay dos empleados con el mismo número de trabajador
  - Hay un empleado sin nombre
  - Aparece una referencia a un departamento que no existe en la relación Departamento.
- Para evitar estos problemas y aumentar la capacidad expresiva del Modelo Relacional, éste se extiende incorporando una serie de Restricciones.
- A este conjunto de restricciones también se les llama Restricciones de Integridad

Empleado

nº_emp	dni	nombre	dirección	dep
1	20.450.120	Juan Pérez	Cuenca 20	d1
2	12.904.569	José Abad	Blasco Ibáñez 35	d2
3	35.784.843	María Gutiérrez	?	d2
4	12.345.678	Pepa Gómez	Colón 15	d2
5	12.345.678	Ana Orts	Cuenca 20	d8
1	45.255.369	?	?	?

Departamento

cod_dep	descripción
d1	Ventas
d2	Compras
d3	Contabilidad

6.1 Restricciones Inherentes

- Son aquellas que no son determinadas por los usuarios, sino que son establecidas por el mismo Modelo Relacional.
- Indican las características propias de una Relación y que han de cumplirse obligatoriamente y que diferencian una Relación de una Tabla.
- Las más importantes son:
  - No puede haber dos tuplas iguales
  - El orden de las tuplas y de los atributos no es relevante
  - Cada atributo sólo puede tomar un valor del dominio al que pertenece
  - Ningún atributo que forme parte de la Clave Primaria puede tomar el Valor Nulo.

6.2 Restricciones Semánticas

- El modelo relacional permite a los usuario incorporar restricciones al modelo para representar la semántica del mundo real.
- Las restricciones semánticas son:
  - Clave primaria (PRIMARY KEY)
  - Unicidad (UNIQUE)
  - Obligatoriedad (NOT NULL)
  - Integridad Referencial (FOREIGN KEY)
  - Verificación (CHECK)
  - Aserción (ASSERTION)
  - Disparador (TRIGGER)

6.2.1 Clave Primaria *(PRIMARY KEY)*

- La restricción de Clave Primaria (PRIMARY KEY) permite declarar un atributo o conjunto de atributos como clave primaria de una relación.
- Características de la Restricción Clave Primaria:
  - No admiten valores nulos (Null)
  - No admiten valores duplicados (Unique)
  - Una relación sólo puede tener asignada una restricción PRIMARY KEY

Clave principal

ProductID	VendorID	AverageLeadTime	StandardPrice	LastReceiptCost
1	1	17	47.8700	50.2635
2	104	19	39.9200	41.9160
7	4	17	54.3100	57.0255
609	7	17	25.7700	27.0585
609	100	19	28.1700	29.5785

Tabla ProductVendor

```
CREATE TABLE Cliente (
CodCliente INT(5) AUTO_INCREMENT PRIMARY KEY,
DNI CHAR(9) UNIQUE,
Nombre VARCHAR(30),
Apellidos VARCHAR(50)
);
```

6.2.2 Unicidad *(UNIQUE)*

- Restricción que permite definir Claves Alternativas o Secundarias.
- Características de la Restricción de Unicidad:
  - No admite valores duplicados
  - Admite valores nulos
  - Esta restricción se puede asignar cuantas veces se precise

```
CREATE TABLE Cliente (
CodCliente INT(5) AUTO_INCREMENT PRIMARY KEY,
DNI CHAR(9) UNIQUE,
Nombre VARCHAR(30),
Apellidos VARCHAR(50)
);
```

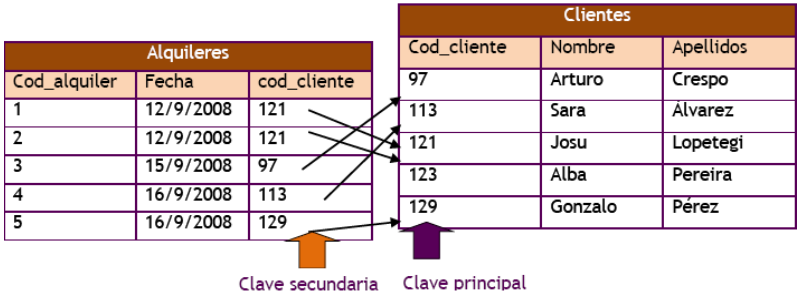
6.2.3 Obligatoriedad *(NOT NULL)*

- Prohíbe que uno o varios atributos de la relación tomen valores Nulos.
- Características de la Restricción de Obligatoriedad:
  - Obliga al usuario a introducir un valor en dicho atributo, en caso contrario impide dar de alta la tupla o registro.
  - Se puede asignar tantas veces como sea necesario

```
CREATE TABLE Cliente (
CodCliente INT AUTO_INCREMENT PRIMARY KEY,
Nombre VARCHAR(30) NOT NULL
Apellidos VARCHAR(50) NOT NULL
);
```

6.2.4 Integridad Referencial *(FOREIGN KEY)*

- La restricción de Clave Ajena se utiliza para enlazar relaciones entre la Clave Principal de una tabla y la Clave Ajena de otra.
- La integridad referencial indica que los valores de la Clave Ajena en la relación hijo o secundaria se corresponden con los de la Clave Primaria en la relación padre o principal.
- Características de la Restricción de Clave Ajena:
  - Los atributos con restricción FOREIGN KEY adminten valores nullos (Null), que indican que dicha tupla no está relacionada con niguna otra de la tabla principal.
  - Evidentemente admite valores repetidos.
  - Esta restricción se puede asignar varias veces a una relación



Ejemplo uso Restricción Clave Ajena (Foreign Key).
Lenguaje SQL - DDL

```sql
create table if not exists Pais
(
    CodPais char(3) primary key,
    Nombre varchar(20),
    NumeroClubs int(4) unsigned,
    Continente set ('Europa', 'América', 'Asia', 'África', 'Oceanía'),
    CodPaisRepresenta char(3),
    constraint fk_CodPaisRepresenta
    foreign key (CodPaisRepresenta) references Pais(CodPais)
);
create table if not exists Hotel
(
    CodHotel char(5) primary key,
    Nombre varchar(20),
    Direccion varchar(45),
    Telefono char(9),
    Email varchar(45) unique,
    Pais char(3),
    Web varchar(45),
    Categoria tinyint(1) unsigned,
    constraint chk_Categoria check (Categoria between 1 and 5),
    constraint fk_Pais foreign key (Pais) references Pais(CodPais)
);
create table if not exists Sala
(
    NumSala int(3) unsigned primary key,
    Capacidad int(4) unsigned,
    Ubicacion varchar(20),
    HotelCodHotel char(5),
    constraint fk_HotelCodHotel foreign key (HotelCodHotel) references Hotel(CodHotel)
);
```

6.2.4.1 Operaciones de Borrado y/o Modificación

- Además de definir las Claves Ajenas hay que tener en cuenta las operaciones de borrado y actualización que se puedan realizar sobre las tuplas de la relación referenciada.
- Sobre el ejemplo Clientes – Alquileres, responde a cuestiones como ¿qué ocurre si eliminase el Cliente cuyo código es el 97? ¿qué ocurre si modifico el código de cliente 97 a 98?
- El Modelo Relacional establece las siguientes posibilidades:
  - Borrado y/o modificación en cascada (CASCADE)
  - Borrado y/o modificación restringido (RESTRICT)
  - Borrrado y/o modificación con puesta a nulos (SET NULL)
  - Borrado y/o modificación con puesta a valor por defecto (SET DEFAULT)

CASCADE

- El borrado o modificación de una tupla en la relación padre (relación con la clave primaria) ocasiona un borrado o modificación de las tuplas relacionadas en la relación hija (la que contiene la clave ajena)
- En el caso de Clientes y Alquiler de la diapositiva anterior:
  - Si se borra un cliente se eliminan todos sus alquileres pertenecientes a dicho cliente
  - Si se modifica un Cod_Cliente de la tabla Clientes, dicha modificación se arrastra a los Alquileres de dicho cliente.

FOREIGN KEY (Cod_Cliente) REFERENCES Clientes(Cod_Cliente)
ON DELETE CASCADE
ON UPDATE CASCADE

RESTRICT

- En este caso no es posible realizar el borrado o modificación de las tuplas de la relación padre si existen tuplas relacionadas con la relación hija.
- En el ejemplo Cliente-Alquiler:
  - No se permite realizar un borrado de un cliente que tiene realizado algún alquiler
  - No se permite modificar un código de cliente que tenga realizado alquiler

FOREIGN KEY (Cod_Cliente) REFERENCES Clientes(Cod_Cliente)
ON DELETE RESTRICT
ON UPDATE RESTRICT

SET NULL

- Esta restricción permite poner la clave ajena en la tabla referenciada a NULL si se produce el borrado o modificación en la tabla primaria o padre.
- En el ejemplo Cliente-Alquiler:
  - Si se borra un cliente que tiene asignado un alquiler se asigna NULL en Cod_cliente de dicho alquiler.
  - Idem si se modifica.

FOREIGN KEY (Cod_Cliente) REFERENCES Clientes(Cod_Cliente)
ON DELETE SET NULL
ON UPDATE SET NULL

SET DAFAULT

- En caso de modificación o borrado de una tupla en la tabla padre, el valor que se pone en las claves ajenas de la tabla referenciada (hija o secundaria) es un valor por defecto establecido a la hora de definir la tabla en el lenguaje de datos SQL.

FOREIGN KEY (Cod_Cliente) REFERENCES Clientes(Cod_Cliente)
ON DELETE SET DEFAULT '0000'
ON UPDATE SET DEFAULT '0000'

Esquema Base de Datos Relacional

id-cliente	*nombre-cliente*	*calle-cliente*	*ciudad-cliente*
19.283.746	González	Arenal	La Granja
01.928.374	Gómez	Carretas	Cerceda
67.789.901	López	Mayor	Peguerinos
18.273.609	Abril	Preciados	Valsaín
32.112.312	Santos	Mayor	Peguerinos
33.666.999	Rupérez	Ramblas	León
01.928.374	Gómez	Carretas	Cerceda

Tabla *cliente*

número-cuenta	*saldo*
C-101	500
C-215	700
C-102	400
C-305	350
C-201	900
C-217	750
C-222	700

Tabla *cuenta*

id-cliente	*número-cuenta*
19.283.746	C-101
19.283.746	C-201
01.928.374	C-215
67.789.901	C-102
18.273.609	C-305
32.112.312	C-217
33.666.999	C-222
01.928.374	C-201

Tabla de relación *cliente-cuenta*

6.2.5 Verificación (CHECK)

- Permite especificar condiciones que deban cumplir los valores de los atributos a los que se les asigne dicha restricción.
- Cada vez que se realice una inserción o actualización de datos se comprueba si los valores cumplen la condición y se rechaza la operación si no se cumple.
- *Ejemplos.*
  - *CHECK (N_HORAS > 30).*
  - *Restringir el campo sueldo para que siempre sea mayor de 1000, sería una regla de validación.*
  - *También por ejemplo que la fecha de inicio sea mayor que la fecha final.*

CodCategoria int(1) check (CodCategoria Between 1 and 5)

6.2.6 Aserción (ASSERTION)

- La aserción funciona de forma similar a la verificación CHECK, pero en este caso la condición puede afectar a varios relaciones.
- La condición se establece sobre elementos de distintas relaciones.

CREATE ASSERTION CONCEDE_SOLICITA AS CHECK (SELECT Cod_Estudiante, Cod_Beca FROM CONCEDE) IN (SELECT Cod_Estudiante, Cod_Beca FROM SOLICITA));

6.2.7 Disparador *(TRIGGER)*

- Las restricciones anteriores son declarativas, sin embargo, este tipo es procedimental.
- El usuario podrá declarar un Procedimiento, Disparador o Trigger, que se ejecutará cuando se ejecute en el sistema una determinada acción sobre una relación, esa acción podrán ser del tipo:
  - INSERT
  - DELETE
  - UPDATE

```
8   delimiter $$
9   DROP TRIGGER IF exists actualizar_cuenta$$
10  CREATE TRIGGER actualizar_cuenta BEFORE INSERT ON movimientos
11  FOR EACH ROW
12  BEGIN
13
14    UPDATE cuentas SET cuentas.saldototal = cuentas.saldototal + NEW.cantidad
15          WHERE cuentas.codcuenta = NEW.codcuenta;
16  END;$$
17
18  DELIMITER ;
19  -- Comprobamos que el disparador se ha creado
20  select * from INFORMATION_SCHEMA.TRIGGERS;
21
22  -- Uso disparador
23  INSERT into movimientos values
24  (null, 1, null, 'I', 200.00),
25  (null, 1, null, 'I', 300.00);
```

7. Vistas

- Una vista de base de datos es el resultado de una consulta SQL.
- Las vistas tienen la misma estructura que una tabla, pero sólo se almacena de ellas su definición, no los datos.
- Sobre una vista se pueden insertar, actualizar, borrar y seleccionar datos, aunque a veces existen restricciones.
- Sobre una BBDD Bancaria podríamos definir una vista con los datos de las cuentas de un determinado cliente. De este modo podríamos permitir que un cliente acceda sólo a sus datos.
- Las vistas son importantes en el diseño de bases de datos cumpliendo tres funciones:
  - Proporcionan un mecanismo de seguridad potente y flexible al ocultar parte de la base de datos a distintos usuarios y aplicaciones
  - Permiten a los usuarios y aplicaciones acceder a los datos de forma personalizada según sus necesidades
  - Permiten simplificar operaciones complejas de consulta, modificación y borrado al poder representarlas con un nombre.

Ejemplos

- *CREATE VIEW [Productos Precio Superior Media] AS*
  SELECT ProductName,UnitPrice
  FROM Products
  WHERE UnitPrice>(SELECT AVG(UnitPrice) FROM Products)

- *CREATE VIEW [Lista Productos Actuales] AS*
  SELECT ProductID,ProductName,Category
  FROM Products
  WHERE Discontinued=No

8. Lenguajes de Datos en el Modelo Relacional

- Ya hemos visto la parte estática del Modelo Relacional, sus componentes.
- Para implementar el Modelo en nuestras aplicaciones necesitamos de una parte dinámica.
- Este conjunto de lenguajes que nos permitan: definir los elementos, establecer las restricciones y manipular los datos.
- Esto genera un conjunto de lenguajes que podemos clasificar según su función:
  - Lenguaje de Definición de Datos
  - Lenguaje de Manipulación de Datos
  - Lenguaje SQL

8.1 Lenguajes Definición de Datos

- Son los llamados DDL (*Data Definition Language*)
- Permiten fundamentalmente:
  - Definición de las Relaciones
  - Restricciones del Modelo
- El más conocido es del DDL de SQL.
- Operaciones típicas:
  - Dar nombre a la Base de Datos o Esquema
  - Declaración dominios
  - Definición de las relaciones individuales del modelo: clave primaria, clave secundaria y clave ajena
  - Definir aserciones y comprobaciones (Checks)
  - Indicar los campos índices de cada relación

8.2 Lenguajes de Manipulación de Datos

- Son los responsables de la parte dinámica del modelo ya que permiten modificar el contenido de las relaciones mediante sentencias.
- Los lenguajes relacionales operan sobre conjunto de tuplas y se dividen en dos tipos:
  - Algebráicos:
    - Los cambios de estado se especifican mediante operaciones cuyos operandos son relaciones y cuyo resultado es otra relación. ***Álgebra Relacional***
    - Operadores del Álgebra Relacional: unión, diferencia y producto cartesiano, más los introducidos por Codd: restricción y proyección.
  - Predictivos:
    - Los cambios de estado se especifican mediante predicados que indican lo que se quiere conseguir sin decir como.
    - Son los lenguajes basados en el ***Cálculo Relacional***
- Los lenguajes de Cálculo Relacional se dividen en dos tipos:
  - Orientados a tuplas QUEL
  - Orientados al dominio como es el QBE

8.3 Lenguaje SQL

- El lenguaje SQL (Structured Query Lenguage, Lenguaje de Consulta Estructurado) es una evolución del lenguajes SEQUEL desarrollado por IBM.
- SQL es normalizado por el Instituto Americano de Normalización (ANSI) y fuen construido en principio como lenguaje algebráico enriquecido con funciones predictivas.
- Contiene un conjunto limitado de palabras claves, distribuidas en tres grandes grupos:
  - DDL (Lenguaje de Descripción de Datos): CREATE, DROP y ALTER
  - DML (Lenguaje de Manipulación de Datos): INSERT, UPDATE, DELETE y SELECT.
  - DCL (Lenguaje de Control de Datos): GRANT, REVOKE, COMMIT, ROLLBACK