1.Introducción al Modelo Relacional.

1.1 ¿Qué es un Modelo ?.

Cuando en teoría de diseño de bases de datos se emplea el término "modelo" , esto no tiene el mismo significado que en Lógica .
En Lógica por "modelo" se entiende una interpretación en la que se satisfacen (se evalúan como ciertas) las fórmulas de una teoría , lo que asegura su consistencia .
En Bases de Datos , un Modelo es un formalismo que nos permite representar la realidad y , más concretamente , aquélla parte de la realidad que nos interesa (Una Empresa , una Universidad , una Biblioteca , el catálogo de las especies protegidas en un determinado país...).
Las partes fundamentales de un Modelo , son :

- Estructura de datos
- Restricciones
- Operadores asociados.

1.2 Estructura de datos en el Modelo Relacional.

El Modelo Relacional fue propuesto por E.Codd en 1970 (E. Codd era entonces un investigador del Centro de IBM en San José (California) y publicó su propuesta en un artículo fundamental que obtuvo el ACM Award correspondiente al Congreso VLDB de 1970) .
La estructura subyacente básica es la **relación** , entendida en su acepción matemática básica : Un subconjunto de un producto cartesiano de conjuntos .

Cuando se pretende describir una parcela de la realidad mediante el formalismo relacional , el primer paso es discernir los **atributos** presentes en el problema .
Un atributo es un ítem elemental de información , en el sentido de no poder desglosarse en componentes más simples.
Los atributos son los nombres que damos a las propiedades de los objetos acerca de los cuales se va a guardar información .

Supongamos que se trata de crear una Base de Datos para una Empresa determinada.
Los datos considerados relevantes por cada empleado , son :

- DNI
- Nombre
- Dirección
- Teléfono
- E-mail
- Puesto
- Antigüedad en la Empresa
- Tipo de salario que se aplica

Estos son por tanto los atributos de empleado relevantes para el problema .

Llamamos **dominio** al conjunto en el cual un atributo toma sus posibles valores .
En nuestro ejemplo :
Dom (DNI) = { cadenas de caracteres alfanuméricos de 9 elementos }
Dom (Nombre)={ cadenas de caracteres alfabéticos de un número máximo de elementos }
Dom (Teléfono)={cadenas de caracteres numéricos de 9 elementos}
Etc…….

En un mismo problema no puede haber dos atributos diferentes con el mismo nombre , pero sí que dos o más atributos diferentes pueden tener dominios idénticos .

En el ejemplo , cada empleado queda descrito por los correspondientes valores de los atributos anteriormente enunciados . Cada empleado se corresponde con una **n-tupla** de valores y el conjunto de las tuplas correspondientes a todos los empleados de la Empresa es la **relación** que podemos denominar “Empleados” por motivos obvios .
Las relaciones se representan en forma tabular , con las columnas etiquetadas por los atributos correspondientes .
Al número de columnas se le denomina “grado” u “orden” de la relación y al número de columnas se le denomina “cardinalidad”.

Empleados

DNI	NOMBRE	DIRECCION	TELEFONO	E_MAIL	PUESTO	ANT.	SAL.
1023R	M.Fernández	Luna 25 Madrid 28025	913367391	mf@pruebas.es	Sales-manager	3	30.000

El ejemplo muestra el contenido de una de las tuplas de la relación “Empleados”. Esta tupla es un elemento del producto cartesiano de los dominios de los diferentes atributos , de modo que podemos afirmar:

Empleados $\subseteq$ Dom(DNI)xDom(NOMBRE)x…xDom(SALARIO)

Una relación es un subconjunto del producto cartesiano de los dominios de los atributos sobre los cuales está definida. Sus elementos son las tuplas .

1.3 Operadores del modelo.

Hay dos tipos de lenguajes relacionales : Los basados en el Cálculo Relacional y los basados en el Algebra Relacional .

El Cálculo Relacional es un subconjunto del Cálculo de predicados de primer orden (sin símbolo de negación y sin símbolos de función) , que tiene la propiedad (demostrada por Codd) de ser relacionalmente completo , es decir : Cualquier información presente en la Base puede ser obtenida formulando una interrogación en términos de Cálculo relacional.
Las interrogaciones son de la forma < meta / condición > , siendo la condición una fórmula cerrada del cálculo .
Estos lenguajes pueden ser orientados a tuplas (ej. : DSL ALPHA) u orientados a dominios (ej. : QBE) , según el tipo de variable que empleen . En todo caso , son lenguajes de alto nivel , no procedimentales , ya que descargan al usuario de la responsabilidad de planificar sus accesos con criterio de optimización de los tiempos de respuesta / ejecución .
La desventaja es que requieren de una familiaridad con la Lógica que no suele tener el usuario “normal”.
El ámbito en que estos lenguajes encuentran su campo natural de aplicación son las Bases de Datos Deductivas , en las que la Lógica es el marco adecuado de descripción uniforme de hechos y reglas , y la repuesta a una interrogación es un caso particular de demostración de teoremas .

El Algebra es un conjunto de operadores que admiten como operandos relaciones y devuelven como resultado relaciones .
Es una ampliación del Algebra convencional de conjuntos (teniendo en cuenta que el elemento de una relación es la tupla) , añadiendo algunos operadores específicos para relaciones .
Es fácil demostrar que todo operador del Algebra puede expresarse en términos del Cálculo , con lo que la completitud relacional de aquélla está también asegurada .
El Algebra es procedimental , así que es importante considerar algunas heurísticas de optimización para minimizar tiempos de respuesta .
El estándar de los lenguajes relacionales , el SQL , está basado en Algebra .

1.4 Restricciones del modelo .

Distinguiremos las **básicas** y las **dependencias .**

Restricciones básicas :

- **Integridad de entidad .**
- **Integridad referencial .**

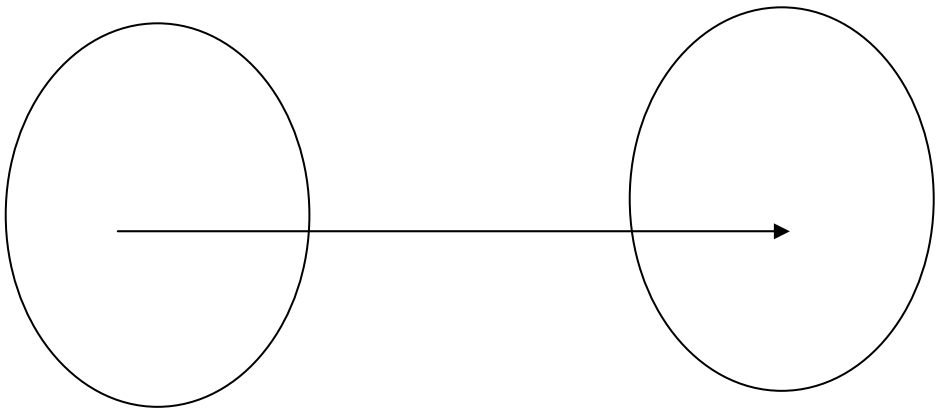
Describiremos estas restricciones una vez introducido formalmente el concepto de **clave** .

Dependencias :

- **Dependencias funcionales**
- **Dependencias multivaluadas**
- **Dependencias de unión**
- **Dependencias generalizadas (proporcionan el marco para la descripción uniforme de los tipos anteriores) .**

Las dependencias funcionales expresan propiedades inherentes a los datos . **Un descriptor (conjunto de atributos) Y depende funcionalmente de otro descriptor X si el valor de X determina unívocamente el de Y y se expresa así : X$\rightarrow$Y (X implica Y) .**

Dom (X) **Dom (Y)**



Una clave es un descriptor mínimo (no redundante) del cual dependen funcionalmente los demás atributos de una relación (en lo sucesivo , para hacer referencia a una estructura de relación con independencia de su contenido , que , en general , variará con el tiempo , usaremos el término " esquema de relación " , o, si no hay ambigüedad , simplemente "esquema") .

En el esquema **Empleados ,** es inmediato comprobar que **DNI** $\rightarrow$ **NOMBRE , DNI** $\rightarrow$ **DIRECCION …DNI** $\rightarrow$ **SALARIO ,** de modo que **DNI** es clave del esquema.

En un esquema puede haber más de una clave . Los atributos que forman parte de alguna clave se llaman principales y los demás no principales .

La **integridad de entidad** establece que ningún atributo principal puede tener valor nulo (desconocido) .

Para ilustrar el concepto de integridad referencial , añadamos a la Base de Datos de ejemplo , una relación de **Proyectos** y otra denominada **Trabajo ,** en la que se detallan los empleados que trabajan en distintos proyectos y las funciones que en ellos desempeñan .

Proyectos

P#	**€**	**Cliente**	**Inicio**	**Fin**
P10	600,000	Montegancedo SA	2003	2004-03

Trabajo.

P#	**DNI**	**Función**
P10	1023R	Director

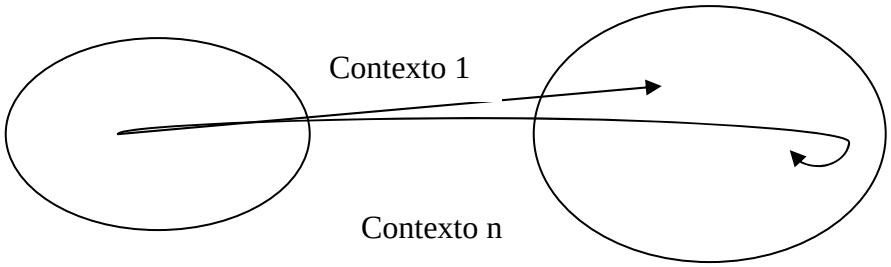
Es fácil ver que la clave de **Trabajo** es **DNI , P# .**
La restricción de referencia obliga a que si (P10,1023R) es un valor de la clave en Trabajo , P10 sea un valor de la clave en Proyectos y 1023R lo sea de DNI en Empleados .

Para concluir esta breve presentación de las restricciones y dependencias , unas notas sobre **Dependencias Multivaluadas .**

Las dependencias multivaluadas (MVD) son de tipo estructural (relativas al contexto) y por ello algo más difíciles de detectar . Si tiene lugar la mvd de Y respecto de X ,
Ello se representa como **X** $\rightarrow\!\!\rightarrow$ **Y.**
Una mvd es una correspondencia entre **Dom(X)** y Π**(Dom(Y))** , de modo que a un valor de X le corresponde un conjunto **bien definido** de valores de Y (independiente de cuáles sean los valores de los demás atributos del mismo

esquema).

Dom (X) Π **(Dom (Y))**



El objetivo de este curso es adquirir la capacidad de diseñar correctamente Bases de Datos Relacionales .
Para ello es imprescindible deducir del análisis del problema las **Dependencias funcionales** , lo que nos permitirá obtener un conjunto de **relaciones normalizadas** a partir de la **Relación Universal** (definida sobre todos los atributos del problema) . Un posterior análisis estructural de las relaciones así obtenidas , nos permitirá detectar posibles mvd's para eventualmente "refinar" el diseño hasta la denominada "Cuarta Forma Normal".
No es objetivo de este curso el estudio de las dependencias de unión ni generalizadas .

1.5 Bibliografía básica .

- **Database and Knowledge-Base Systems**
  J.D.Ullman
  Computer Science Press , 1988
- **The Theory of relational Databases**
  D.Maier
  Computer Science Press , 1983
- **Relational Databases**
  Chao-Chih Yang

Prentice Hall , 1986