Departamento de Lenguajes y Ciencias de la Computación
Universidad de Málaga

Administración de Bases de Datos

(Ingeniería Técnica en Informática de Gestión)

Sistemas de Bases de Datos:
Introducción y Conceptos Básicos para la Administración y de Bases de Datos Relacionales

E.T.S.I. Informática ***J. Galindo Gómez***

Introducción: Los SGBD

- **SGBD** (Sistema Gestor de Bases de Datos) o **DBMS** (*DataBase Management System*)**:**
  - Un **SGBD** está formado por:
    - **Base de Datos (BD):** Colección de datos sobre un ente particular (empresa, organización, tema...).
    - **Programas (*Software*)** para acceder y manipular esos datos.
    - Otros elementos que pueden considerarse: ***Hardware*** **y Usuarios.**
  - Los **SGBD** surgen, propiamente, a mediados de los años 60's, como un intento de mejorar el **Sistema de Procesamiento de Archivos** utilizado hasta entonces, de forma que una base de datos sea Integrada y Compartida:
    - **Integrada:** Unificación de archivos e información (evitando redundancias, mejorando los accesos...).
    - **Compartida:** Información utilizada por varios usuarios (total o parcialmente).

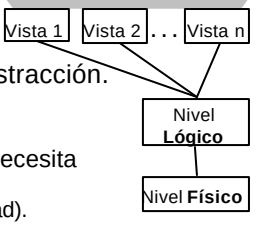
Introducción: Los SGBD

- **Sistema de Procesamiento de Archivos:** Tiene una serie de *inconvenientes* que son reducidos en los SGBD:
  - **Redundancia** (datos duplicados) e **Inconsistencia** (datos contradictorios).
  - **Dificultad de Acceso a ciertos datos o información:** Si no existen programas para acceder o calcular cierta información, no puede accederse a ella. Ej.: Calcular totales, o registros con cierta condición...
  - **Aislamiento de Datos:** Los datos pueden estar en varios archivos con distintos formatos, que complican la creación de programas nuevos.
  - **Falta de Integridad:** Es complicado mantener ciertas condiciones en la información. Ej.: Que el saldo sea superior a cierta cantidad, que un empleado no esté adscrito a un número de Departamento que no exista...
  - **Problemas de Atomicidad en las operaciones:** A veces es esencial que para la consistencia de la BD se efectúen varias operaciones como si fueran una única operación, evitando que se produzcan fallos en medio de dicha operación. Ej.: En una transferencia bancaria hay que quitar dinero de una cuenta y añadirlo a la otra.
  - **Problemas en el Acceso Concurrente:** Si varios usuarios acceden a la vez a un dato pueden producirse errores. Ej.: Si se saca dinero de una misma cuenta desde dos sitios distintos.
  - **Problemas de Seguridad:** Dificultad para controlar que ciertos usuarios no accedan a ciertos datos.

Introducción: Los SGBD

- **Independencia de los Datos:** Puede ser Física o Lógica.
  - Es una de las grandes ventajas de los SGBD.
  - **Objetivo: Que las aplicaciones no dependan de cómo se almacenen los datos físicamente o de su organización lógica**.
    - Puede modificarse la estructura de los datos sin que afecte a los programas. Ej.: Modificar nombres y formato de los ficheros, añadir o borrar campos...
    - El **DBA** (*DataBase Administrator*) puede modificar esa estructura de los datos para mejorar el acceso y optimizar el uso de los recursos: Añadir o borrar índices, dividir en distintos archivos, usar nuevos discos u ordenadores (bases de datos distribuidas)...
    - El concepto y las ventajas son similares al uso de TDA en programación.
  - Esto implica que las aplicaciones (o los usuarios) tienen una **Visión Abstracta** de los datos: El SGBD esconde detalles sobre cómo se almacenan y mantienen los datos, de forma que facilita el acceso a estos datos, ya que sólo es necesario pedir al SGBD qué datos se requieren, sin especificar cómo conseguirlos.
    - Este es el principal objetivo de la **Arquitectura ANSI/SPARC** de 3 niveles.
      - La idea básica es poner un nivel intermedio entre las aplicaciones (o los usuarios) y el sistema de almacenamiento físico.

Informe ANSI/SPARC (1978): 3 Niveles



- **1. NIVEL EXTERNO (o de Vistas):** Mayor nivel de abstracción.
  - Es el nivel más cercano a los usuarios y a su forma de ver los datos (su punto de vista y sus necesidades).
  - Cada nivel externo describe sólo la parte de la BD que necesita para su aplicación o uso (de un programa, un usuario...).
    - Esto facilita el control de los permisos de acceso (seguridad).
  - Existen distintos tipos de usuarios: Programador en C (ve los datos como estructuras struct), usuario final de un programa...
- **2. NIVEL CONCEPTUAL (o Lógico):** Conecta los otros dos niveles.
  - Describe qué datos se almacenan en la BD y qué relaciones existen entre ellos.
  - Los programadores de BD trabajan con este nivel, aunque también tienen su vista particular. Nunca un programa debe acceder directamente al nivel interno.
  - **Independencia Lógica de los Datos:** Cuando modificar este nivel no implica la modificación de los niveles externos (programas).
    - En muchas ocasiones la dependencia lógica de los programas es fuerte.
- **3. NIVEL INTERNO (o Físico):** El más cercano al almacenamiento físico.
  - Describe cómo se almacenan realmente los datos (estructuras de datos complejas de bajo nivel: Registros, ficheros, bytes, datos, organización...
  - Los DBA pueden conocer y controlar ciertos aspectos de este nivel, pero no es necesario: Puede conseguirse optimizar el sistema y ganar en eficiencia (*tuning*).
  - **Independencia Física de los Datos:** Cuando modificar este nivel no implica la modificación de los niveles externos (programas).

Clasificación de los SGBD

- Dependiendo del **Modelo de Datos** utilizado:
  - Modelo **Relacional**: Un conjunto de tablas.
  - Modelo **Dirigido a Objetos**: Un conjunto de objetos. Una clase es un tipo de objeto con las mismas características (o propiedades) y operaciones (o métodos). Las clases forman una jerarquía (grafo acíclico).
  - Modelo **Relacional dirigido a Objetos**.
  - Otros modelos antiguos: **Jerárquico** y en **Red** (CODASYL).
- Dependiendo del **Número de Usuarios** que soporta:
  - **Monousuario** y **Multiusuario**.
- Dependiendo del **Número de Sitios** en los que está la BD:
  - **SGBD Centralizado**.
  - **SGBD Distribuido** (y conectado mediante una red): Pueden ser **homogéneos** (con mismo software en todos los lugares) o **heterogéneos**. Esto genera SGBD **Federados** (*Multidatabase)*, donde existen diversos SGBD autónomos con cierto acoplamiento (o comunicación) entre ellos.
- Otros: **ODBC** (*Open Database Connectivity*), estándar de comunicación con SGBD.
  - De **propósito general** o **específico**.
  - **Sistemas OLTP** (*On-Line Transaction Processing*): Sistemas con muchas transacciones concurrentes y necesitando rapidez (pocos retardos).

Gestión de Bases de Datos

- **ESQUEMA de la BD:** Diseño completo de la BD a nivel conceptual (datos, relaciones entre ellos, restricciones básicas...).
  - Una vez definido el esquema de una BD no es usual cambiarlo.
  - El contenido de una BD (dentro de cierto esquema) es habitual modificarlo, puesto que el mundo cambia. Por eso se dice que un esquema puede tener varios **ejemplares, instancias** (*database instance*) o **estados** (*database state*), que tendrán valores distintos en sus datos.
- **Lenguajes de Gestión de BD: SDL, DDL y DML.**
  - **SDL** (*Storage Definition Languaje*) o **Lenguaje de Definición del Almacenamiento:**
    - Son comandos para definir ciertos parámetros para el **almacenamiento Físico:**
      - Estructura de los ficheros, índices de acceso, agrupar tablas afines...
    - **No son de SQL** porque no son del nivel del esquema conceptual.
    - Cada **SGBD** puede tener distintos comandos.
    - Algunos ejemplos para el SGBD **Oracle:**
      - **`CREATE, ALTER y DROP TABLESPACE.`**
      - **`CREATE, ALTER y DROP INDEX.`**
      - **`CREATE, ALTER y DROP CLUSTER.`**
      - **`CREATE, ALTER y DROP ROLLBACK SEGMENT.`**

Gestión de Bases de Datos

  - **DDL (*Data Definition Language*) o Lenguaje de Definición de Datos (LDD):** Se utiliza para definir el ESQUEMA de la BD.
    - Esa información se almacena en el **Diccionario de Datos** (metadatos)**.**
    - **Ejemplos en SQL: `CREATE, ALTER y DROP <Objeto>`,** donde el objeto puede ser**: `TABLE, VIEW, MATERIALIZED VIEW...`**
  - **DML (*Data Manipulation Language*) o Lenguaje de Manipulación de Datos (LMD):** Se utiliza para efectuar operaciones sobre los datos:
    - **Recuperaciones o Consultas a la BD (*retrievals, select*).**
      - Sentencia **`SELECT`** de **SQL**.
      - Los Lenguajes de Consulta pueden ser **Procedimentales** (hay que especificar cómo obtener los datos) y **No Procedimentales** (hay que especificar qué datos se desean y no cómo obtenerlos).
        - » En general los Procedimentales son más eficientes.
    - **Actualizaciones o Modificaciones a la BD (*updates*).**
      - **Insertar (*Insert*):** Introducir nuevos datos en la BD.
        - » Sentencia **`INSERT`** de **SQL**.
      - **Borrar (*Delete*):** Borrar datos de la BD.
        - » Sentencia **`DELETE`** de **SQL**.
      - **Modificar (*Update*):** Modificar valores de datos ya existentes.
        - » Sentencia **`UPDATE`** de **SQL**.

Gestión de Transacciones

- **TRANSACCIÓN:** Conjunto de operaciones de una aplicación de Bases de Datos que se efectúan como una única operación lógica.
  - **Atomicidad:** Una transacción se considera como una única operación lógica, aunque esté formada por varias operaciones más simples.
    - Debe controlarse que una transacción no sea ejecutada "a medias".
    - Una transacción, o se ejecuta completamente o no se ejecuta.
  - **Consistencia:** Antes y después de cada transacción, la BD debe tener valores lógicos, aceptables o consistentes.
    - Si se produce un fallo del sistema en medio de la transacción, se deben anular las primeras operaciones de la transacción ya efectuadas: Los valores de la BD deben ser **consistentes**.
    - La consistencia de la BD debe asegurarse incluso aunque varias transacciones se efectúen concurrentemente.
  - **Durabilidad:** Tras una transacción, los nuevos valores de la BD deben mantenerse a pesar de cualquier tipo de fallo del sistema.
- En sistemas de BD pequeños pueden no cumplirse todos estos requisitos o pueden evitarse los problemas que generan (por ejemplo, prohibiendo transacciones concurrentes).

Funciones del DBA (administrador)

- **Control Global de la BD:** El DBA es en general el que controla el funcionamiento global de la BD. Puede ser una o varias personas.
- **Definir el Esquema de la BD:** A partir del modelo conceptual, construye sentencias del DDL que se traducirán en estructuras de datos e información en el diccionario de datos del SGBD usado.
- **Definir Estructuras de Almacenamiento:** Se trata de definir cómo y donde se almacenará cada tipo de datos: Datos de la BD y del diccionario, del sistema de recuperación (*rollback system*), de los índices (*indexes*), de las aplicaciones o herramientas (*tools*), de información temporal...
  - La correcta administración de estas estructuras de almacenamiento puede conseguir aumentar mucho la **eficiencia de un SGBD**, lo cual es muy importante.
- **Conceder/Revocar Permisos de Acceso** (*grant/revoke*)**:** Para consultar, insertar, actualizar o borrar datos, pero también para crear objetos (tablas, vistas, índices, usuarios...), ejecutar o crear programas, conceder permisos a terceros...
- **Especificación de Restricciones de Integridad:** Para no admitir valores imposibles o repetidos en la BD, restric. de integridad referencial...

Usuarios de la BD

- **Programadores de Aplicaciones:** Programan aplicaciones de uso más o menos específico o general en un lenguaje de programación de alto nivel. Pueden distinguirse dos Tipos de Aplicaciones:
  - **Aplicaciones Internas al SGBD:** Utilizan un lenguaje de programación que admita el SGBD y permiten la creación de programas que se almacenan y ejecutan "dentro" del SGBD.
    - Este tipo de programas puede ser muy variado, como por ejemplo:
      - Disparadores (*triggers*) que se ejecutan si se cumple cierta condición.
      - Funciones para usarlas en las sentencias del DML (como en la sentencia SELECT de SQL).
      - Procedimientos para cualquier acción, que pueden ser utilizados por otros usuarios (directamente o a través de otro programa).
    - Ejemplo de este tipo de lenguajes es el PL/SQL del SGBD Oracle.
  - **Aplicaciones Externas al SGBD:** Utilizan un lenguaje externo al SGBD (C/C++, Pascal, Cobol...) y dentro de este lenguaje anfitrión se utilizan llamadas especiales del lenguaje DML (como SQL inmerso, *embedded*) que son traducidas por un *precompilador* a sentencias del lenguaje anfitrión.
    - Existen **lenguajes de cuarta generación** que facilitan el control de los objetos de una BD (manipulación, mostrar los datos en una plantilla de datos, generación de formularios...). Por ejemplo, existe el Developer 2000 del SGBD Oracle.
- **Usuarios Sofisticados y Especializados:** Interactúan con el SGBD sin usar programas intermedios, sino directamente con el DML del SGBD. También pueden escribir sus propios programas para los más variados fines (CAD, *Data Mining*, Sistemas Expertos...).
- **Usuarios Normales:** Interactúan con el SGBD a través de alguno de los programas de aplicación. Pueden ser programas muy generales y potentes o muy específicos (como un cajero automático).

Componentes de un SGBD

- **Componentes de Procesamiento de Sentencias DML/DDL:**
  - **Compilador DML:** Traduce las sentencias DML en instrucciones a bajo nivel para ser ejecutadas por el Motor de Ejecución del SGBD. Suelen usar algoritmos para optimizar las sentencias DML y hacerlas más eficientes.
  - **Precompilador DML:** Traduce las sentencias DML incrustadas en cierto programa en sentencias propias de su lenguaje de programación.
  - **Intérprete DDL:** Interpreta las instrucciones DDL y genera los metadatos necesarios en el diccionario de datos.
  - **Motor de Ejecución:** Encargado de que se ejecuten las sentencias ya compiladas, utilizando los componentes de Gestión de Almacenamiento.
- **Componentes de Gestión de Almacenamiento:** Se encargan del acceso directo a los datos solicitados por el Motor de Ejecución.
  - **Gestor de Autorización e Integridad:** Comprueba que el usuario tiene los permisos pertinentes y que no se violan las restricciones de integridad.
  - **Gestor de Transacciones:** Asegura un estado consistente de la BD, aunque se produzcan fallos del sistema o existan transacciones concurrentes.
  - **Gestor de Archivos:** Gestiona el espacio en disco a través de archivos.
  - **Gestor de Memoria Intermedia:** Gestiona los datos que deben traerse/llevarse de disco a Memoria principal.
- **Otros Componentes Almacenados en Disco:** Archivos de datos, diccionario de datos, índices, datos estadísticos...

Modelo de DATOS RELACIONAL

- Introducido por Ted **Codd** de IBM, en 1970.
- Se basa en los conceptos de relación matemática, teoría de conjuntos y en la lógica de predicados de primer orden.
- **Base de Datos Relacional** (**BDR**) **=** Conjunto de Relaciones.
  - **Relación o Tabla** (*relation, table*)**:** **Lista de valores** con un nombre, donde cada valor es una fila (registro), compuesto por 1 o más columnas (campos).
    - **Fila o Tupla** (*row, tuple*): Hecho que corresponde a una entidad o relación en el mundo real. Sin repeticiones.
    - **Columna o Atributo** (*column, attribute*): Valor relacionado con ese hecho, sobre un aspecto particular.
      - Todos los valores de una columna son del **mismo tipo o dominio**.
      - Los valores (y el dominio) deben ser **atómicos o indivisibles** (como información). Ejemplos: Números naturales, reales, cadenas de caracteres...
      - **Formato:** Forma de representar un dato. Ej: Tlfno: +34 999-99-99-99.
        - » Una fecha también puede representarse de muchas formas.

Persona

Nombre	Altura	Peso
Santiago	1.79	78
Jesús	1.82	80
María	Null	61

INTENSIÓN y EXTENSIÓN de una BDR

- **INTENSIÓN o Esquema de una Relación:** Características casi invariables de la relación: **$R(A_1, A_2, ... A_n)$**
  - Nombre: R.
  - Grado (número de atributos): n
  - Nombre de los atributos: A_1, A_2, ... A_n.
  - Dominio de cada atributo: $dom(A_1)$, $dom(A_2)$, ... $dom(A_n)$.
    - $\forall i=1,2,...$ N: Null $\in dom(A_i)$
  - Ejemplo: **Persona(Nombre,Altura,Peso).**
- **EXTENSIÓN o RELACIÓN:** Valores de las tuplas, los cuales pueden variar con el tiempo.
  - Una relación r con esquema R, se representa como r(R) y es un conjunto de tuplas de la forma $\{t_1, ... t_m\}$, con $t_i = \{v_1, v_2, ... v_n\}$, tal que, $v_j \equiv t_i[A_j] \in dom(A_j)$.
  - Por tanto: $r(R) \subseteq (dom(A_1) \times dom(A_2) \times ... \times dom(A_n))$.
  - Con dominios finitos, el máximo número de tuplas es: $|dom(A_1)| * |dom(A_2)| * \cdots * |dom(A_n)|$

CARACTERÍSTICAS de las Relaciones

- **No existe orden entre las tuplas**, aunque lógicamente se almacenen y se visualicen en un cierto orden, pero ese orden puede variar sin que suponga alterar la relación.
- **No existe orden entre los atributos** y, por tanto, puede cambiarse el orden sin que suponga un cambio a la relación. Por comodidad suele suponerse un orden en los atributos y así se representarán los valores de las tuplas en ese orden.
- **Primera Forma Normal:** Los valores son atómicos o indivisibles. Valores compuestos o multivaluados no son permitidos.
  - **Atributos Compuestos:** Se representan sólo sus componentes simples.
  - **Atributos Multivaluados:** Se representa cada atributo multivaluado en una relación separada. Dicha relación almacenará todos los valores.
- **Interpretación de una Relación:**
  - El esquema de una relación es un aserto o afirmación que indica los atributos del objeto (entidad o relación) que representa la relación.
  - Cada tupla es un hecho o instancia de ese objeto, que puede representarse como un predicado.

NOTACIÓN del MODELO RELACIONAL

- **Esquema de una Relación de grado n: $R(A_1, A_2, ... A_n)$.**
- **Relación r con el esquema R: r(R).**
- **n-tupla t de r(R) con n valores componentes:**

  $$t = <v_1, v_2, ...v_n>, \quad v_i \hat{I} A_i.$$

  - Valor v_i en t del atributo A_i: $t[A_j]$ º $t.A_j$.
  - Subtupla con algunos valores de t: $t[A_u, ... A_z]$ º $t.(A_u, ... A_z)$.
- **Atributo A de la relación R: R.A**
  - Así se diferencian atributos con igual nombre de distintas relaciones: R.A, S.A, Q.A...
  - No pueden existir dos atributos con el mismo nombre en una misma relación.
- Para evitar problemas en la implementación, los nombres de Atributos y Relaciones no deben contener espacios en blanco ni caracteres “raros” (acentos, ñ, Ñ...).

RESTRICCIONES RELACIONALES

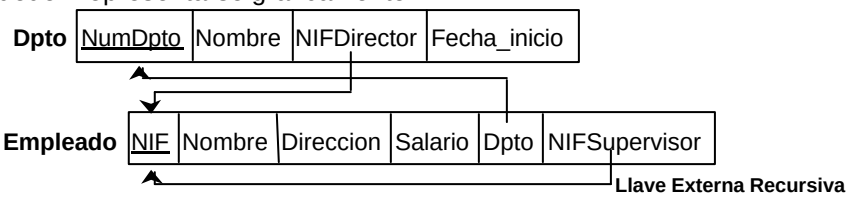
- **Restricciones de DOMINIO:** Los valores deben pertenecer a su dominio correspondiente y ser atómicos. Puede restringirse el valor **Null.**
- **Restricciones de LLAVE:** En una relación no hay tuplas repetidas. Por tanto, hay un conjunto de atributos SK tal que no existen 2 tuplas con iguales valores en SK: $\forall t_1,t_2 \in r(R): t_1 \neq t_2 \Rightarrow t_1[SK] \neq t_2[SK]$
  - **Superllave o Superclave** (*superkey*): Cualquier conjunto de atributos SK que cumple la **restricción de unicidad:** Que no existan dos tuplas con el mismo valor en los atributos de SK.
    - Todos los atributos de una relación forman siempre una superllave.
  - **Llave o Clave:** Superllave mínima, sin atributos superfluos. Si se quita un atributo de la llave, deja de ser superllave.
    - **Llave Candidata:** Llave que existe en una relación (pueden existir varias).
    - **Llave Primaria:** Llave Candidata que se usará para identificar las tuplas de la relación. Suele escogerse la que tenga menos atributos. Se representará subrayada. Ejemplo:
      Estudiante(NIF, Nombre, Fecha_Nacimiento, Direccion, Telefono);
    - **Llave Externa:** Atributos de una relación que son Llave Primaria en otra.
      - Sirve para enlazar una relación con otra.

BASES de DATOS RELACIONALES

- **Esquema de BDR:**
  - Formado por:
    - Conjunto de **Esquemas de Relación S: S={R_1, R_2, ... R_n}.**
    - Conjunto de **Restricciones de Integridad**, para ser cumplidas.
  - Un esquema puede tener varios **ejemplares, instancias** (*database instance*) o **estados** (*database state*), con valores distintos en sus relaciones. Cada uno de ellos forma una **extensión** de la relación.
- **Restricciones de Integridad:**
  - **Integridad de Entidad:** Una llave primaria no puede ser Null.
  - **Integridad Referencial:** Si en una relación hay una referencia a una tupla de otra relación, esa referencia debe ser a una tupla que exista.
    - **Ejemplo: Dpto** (NumDpto,Nombre,NIFDirector…)
      - NIFDirector indica el NIF del director del departamento número NumDpto. Así, NIFDirector, debe existir en la relación **Empleado**(NIF,Nombre,Direccion...).
      - NIFDirector en la relación **Dpto** es una **LLAVE EXTERNA**.

BASES de DATOS RELACIONALES

- **Las Restricciones de Integridad Referencial (llaves externas...)** pueden representarse gráficamente**:**



- Los **SGBD** tienen mecanismos para forzar a que se cumplan las restricciones de integridad que se establezcan.
- **Restricciones de Integridad Semánticas:** Dependen del contexto.
  - Ejemplos: Salario **>** 0, Número de horas semanales trabajadas en un proyecto **<** 50, SalarioSupervisor **>** SalarioSupervisado…
  - Se controlan mediante disparadores (***triggers***) y condiciones (***assertions***).
- **Rest. de Estado:** Todas las ya vistas (Referencial y de Entidad).
  - Hacen que la BD sea consistente.
- **Rest. de Transición:** Se aplican en las modificaciones y dependen del estado anterior. Se controlan mediante *triggers*.
  - Ejemplo: El salario sólo puede aumentar.

INSERTAR en la BD: Restricciones

- **INSERTAR una tupla t en R (INSERT):**
  - Puede violar las restricciones siguientes:
    - **De Dominio:** Si un valor no pertenece al dominio que debe.
    - **De Llave:** Si la llave de la tupla ya existe en R.
    - **De Entidad:** Si la llave primaria vale Null.
    - **De Integridad Referencial:** Si una llave externa no existe en la relación referenciada y tiene un valor distinto de Null.
  - En ese caso, se pueden efectuar las siguientes acciones:
    - **Rechazar la inserción**, indicando el motivo.
    - **Solicitar** la **información** necesaria para subsanar la violación.

BORRAR de la BD: Restricciones

- **BORRAR una tupla t en R (DELETE):**
  - Sólo puede violar la restricción **de Integridad Referencial:** Si se borra una tupla que es referenciada en otra relación.
  - En ese caso, se pueden efectuar las siguientes acciones y el SGBD debería permitir al usuario elegir cual es la más apropiada para cada caso (cada llave externa):
    - **Rechazar el borrado**, indicando el motivo.
    - **Propagar el borrado**, borrando todas las tuplas que hagan referencia a la que se quiere borrar. El borrado de esas tuplas puede generar el borrado en cascada de otras.
    - **Modificar los valores** que hacen referencia a la tupla borrada: O se ponen otros valores válidos (valores por defecto, por ejemplo) o se establecen a Null (si no son parte de la llave primaria).

ACTUALIZAR en la BD: Restricciones

- **ACTUALIZAR valores de una tupla t en R (UPDATE):**
  - Puede violar las restricciones siguientes:
    - **De Dominio:** Si el nuevo valor no pertenece al dominio que debe.
    - **De Llave:** Si al modificar la llave de la tupla ya existe en R.
    - **De Entidad:** Si se actualiza la llave primaria al valor Null.
    - **De Integridad Referencial:**
      - Si al modificar una llave externa con un valor distinto de Null, ese valor no existe en la relación referenciada.
      - También, si modificamos una llave primaria que es referenciada como llave externa en otra relación.
  - En ese caso, existen las siguientes acciones:
    - **Rechazar la actualización**, indicando el motivo.
    - **Solicitar** la **información** necesaria para subsanar la violación.
    - Si se incumple la restricción de integridad referencial porque estamos modificando una llave primaria podemos optar por **propagar dicha modificación, o bien establecer los valores** de las llaves externas a **Null** o a un **valor por defecto**.