

Introducción a las Bases de Datos Relacionales

Máster en Data Science

M1967 - Modelos de Datos y Sistemas de Información 2019-2020

Bases de datos relacionales: Gestores

- Un Sistema Gestor de Bases de Datos Relacionales (en adelante, SGBDR) es un programa, o conjunto de programas, que, siguiendo el modelo relacional, permiten el acceso, manipulación, almacenamiento y otras gestiones sobre los datos, garantizando su persistencia y su accesibilidad.
- Almacenan los datos de forma estructurada y normalizada, permitiendo definir restricciones y mecanismos que garanticen la consistencia, integridad y seguridad en el acceso. Además, garantizan que las transacciones que manipulan los datos cumplen con el criterio ACID.
 - Hablaremos de todo esto a continuación.

Bases de datos relacionales

- En una base de datos relacional podemos encontrar los siguientes elementos:
 - **Entidades:** representan “algo” del mundo real sobre el que queremos almacenar sus datos dentro de un contexto. En el modelo relacional, se representan como **tablas**.
 - En una base de datos de una universidad (contexto), algunas posibles entidades (tablas) serían los profesores, los planes de estudio, las asignaturas, los estudiantes, el personal de administración, las facultades...

ATENCIÓN: SIEMPRE EN SINGULAR

Profesor

Plan

Asignatura

Facultad

Bases de datos relacionales

- En una base de datos relacional podemos encontrar los siguientes elementos:
 - **Atributos o campos:** son características de las entidades que deseamos almacenar.
 - Por ejemplo, los profesores pueden tener almacenado un identificador, su nombre y apellidos o su fecha de nacimiento, las asignaturas su identificador, nombre o créditos, etc.

Identificador	Nombre	Apellidos	...
Uc0001	Diego	García Saiz	...
Uc0002	Valvanuz	Sierra	...
...

Tabla Profesor

Identificador	Nombre	...
m301	Modelado de Datos	...
m389	Minería de datos	...
...

Tabla Asignatura

Bases de datos relacionales

- En una base de datos relacional podemos encontrar los siguientes elementos:
 - **Atributos, campos o columnas:** son características de las entidades que deseamos almacenar.
 - Por ejemplo, los profesores pueden tener almacenado un identificador, su nombre y apellidos o su fecha de nacimiento, las asignaturas su identificador, nombre o créditos, etc.
 - En un SGBDR, cada campo ha de ser de un **tipo** concreto. Por ejemplo, la fecha de nacimiento de un profesor podrá ser de tipo fecha, su nombre de tipo string o char, o los créditos de las asignaturas de tipo numérico (entero, real,...).
 - En un SGBDR, los campos además pueden tener **restricciones**. Por ejemplo, puede indicarse si es obligatorio o no (el segundo apellido de un profesor), o un rango de valores posible (no puede haber asignaturas con créditos negativos), entre otros.

Bases de datos relacionales

- **Atributos, campos o columnas** : son características de las entidades que deseamos almacenar.
 - En el modelo relacional, habrá al menos un atributo (o conjunto de atributos) que identifiquen de forma única a cada instancia de una tabla. Por ejemplo, el identificador es un campo único para cada uno de los profesores. A este campo se le denomina **PRIMARY KEY**.
 - Este campo sirve para que otras tablas lo referencien. Por ejemplo, en la tabla asignaturas podemos encontrar un campo que contiene el identificador del profesor responsable. Este campo es conocido como **FOREING KEY**.
 - Cuando existan varios campos con valor único en la tabla, estos se denominarán **CANDIDATE KEY**. Uno de ellos será elegido como PRIMARY KEY, y los otros quedarán como **ALTERNATE KEY**.

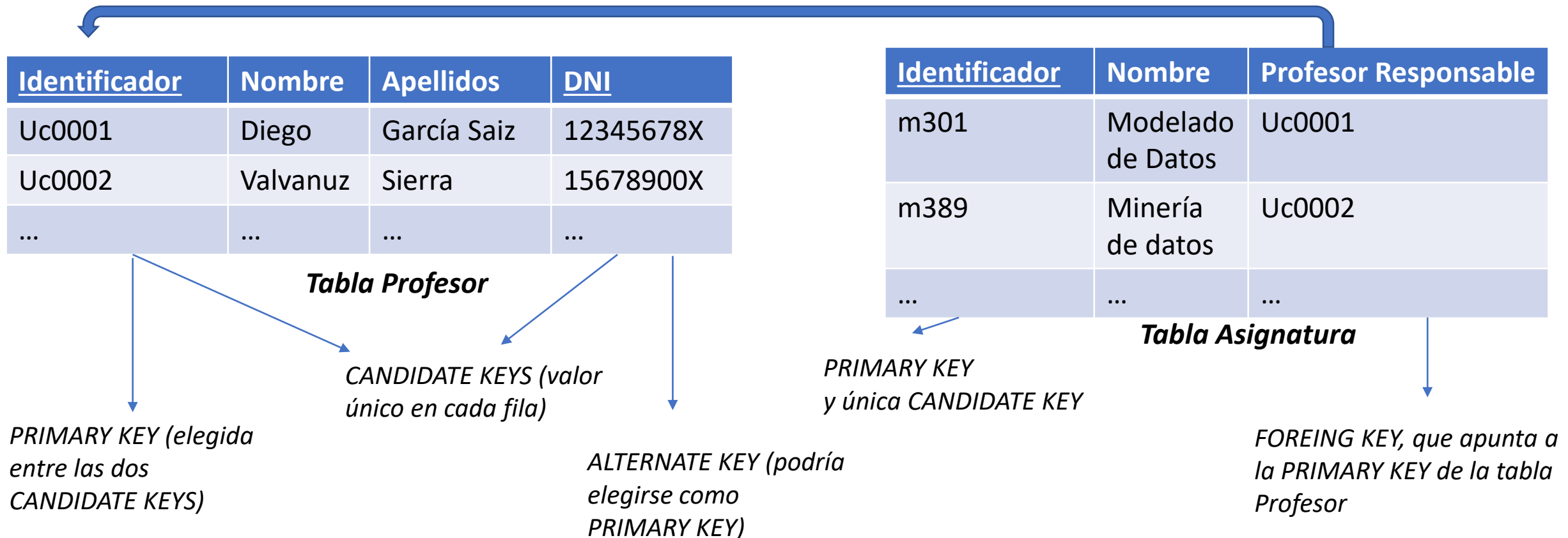
Identificador	Nombre	Apellidos	DNI
Uc0001	Diego	García Saiz	12345678X
Uc0002	Valvanuz	Sierra	15678900X
...

Identificador	Nombre	Profesor Responsable
m301	Modelado de Datos	Uc0001
m389	Minería de datos	Uc0002
...

¿Cuáles son las diferentes “KEYs” de estas tablas?

Bases de datos relacionales

- **Atributos, campos o columnas** : PRIMARY KEY (PK), FOREIGN KEY (FK), CANDIDATE KEY (CK) Y ALTERNATE KEY (AK)



Bases de datos relacionales

- **Atributos, campos o columnas** : PRIMARY KEY (PK), FOREIGN KEY (FK), CANDIDATE KEY (CK) Y ALTERNATE KEY (AK)

SÓLO PUEDE DEFINIRSE UNA PRIMARY KEY, EL RESTO QUEDAN COMO ALTERNATE KEYS

<u>Identificador</u>	Nombre	Apellidos	<u>DNI</u>
Uc0001	Diego	García Saiz	12345678X
Uc0002	Valvanuz	Sierra	15678900X
...

Tabla Profesor

CANDIDATE KEYS (valor único en cada fila)

PRIMARY KEY (elegida entre las dos CANDIDATE KEYS)

ALTERNATE KEY (podría elegirse como PRIMARY KEY)

<u>Identificador</u>	Nombre	Profesor Responsable
m301	Modelado de Datos	Uc0001
m389	Minería de datos	Uc0002
...

Tabla Asignatura

PRIMARY KEY y única CANDIDATE KEY

FOREING KEY, que apunta a la PRIMARY KEY de la tabla Profesor.

Bases de datos relacionales

- **Atributos, campos o columnas** : PRIMARY KEY (PK), FOREIGN KEY (FK), CANDIDATE KEY (CK) Y ALTERNATE KEY (AK)

UN CAMPO FOREIGN KEY DE UNA TABLA APUNTA A OTRO CAMPO PRIMARY KEY O ALTERNATIVE KEY DE OTRA TABLA. NO PUEDE APUNTAR AL RESTO DE CAMPOS. ESTE DISEÑO ESTÁ BIEN, APUNTA A LA PRIMARY KEY

<u>Identificador</u>	Nombre	Apellidos	<u>DNI</u>
Uc0001	Diego	García Saiz	12345678X
Uc0002	Valvanuz	Sierra	15678900X
...

Tabla Profesor

CANDIDATE KEYS (valor único en cada fila)

PRIMARY KEY (elegida entre las dos CANDIDATE KEYS)

ALTERNATE KEY (podría elegirse como PRIMARY KEY)

<u>Identificador</u>	Nombre	Profesor Responsable
m301	Modelado de Datos	Uc0001
m389	Minería de datos	Uc0002
...

Tabla Asignatura

PRIMARY KEY y única CANDIDATE KEY

FOREING KEY, que apunta a la PRIMARY KEY de la tabla Profesor

Bases de datos relacionales

- **Atributos, campos o columnas** : PRIMARY KEY (PK), FOREIGN KEY (FK), CANDIDATE KEY (CK) Y ALTERNATE KEY (AK)

UN CAMPO FOREIGN KEY DE UNA TABLA APUNTA A OTRO CAMPO PRIMARY KEY O ALTERNATIVE KEY DE OTRA TABLA. NO PUEDE APUNTAR AL RESTO DE CAMPOS. ESTE DISEÑO ESTÁ BIEN, APUNTA A UNA ALTERNATE KEY, QUE TAMBIÉN ES ÚNICA

<u>Identificador</u>	Nombre	Apellidos	<u>DNI</u>
Uc0001	Diego	García Saiz	12345678X
Uc0002	Valvanuz	Sierra	15678900X
...

Tabla Profesor

CANDIDATE KEYS (valor único en cada fila)

PRIMARY KEY (elegida entre las dos CANDIDATE KEYS)

ALTERNATE KEY (podría elegirse como PRIMARY KEY)

<u>Identificador</u>	Nombre	Profesor Responsable
m301	Modelado de Datos	12345678X
m389	Minería de datos	15678900X
...

Tabla Asignatura

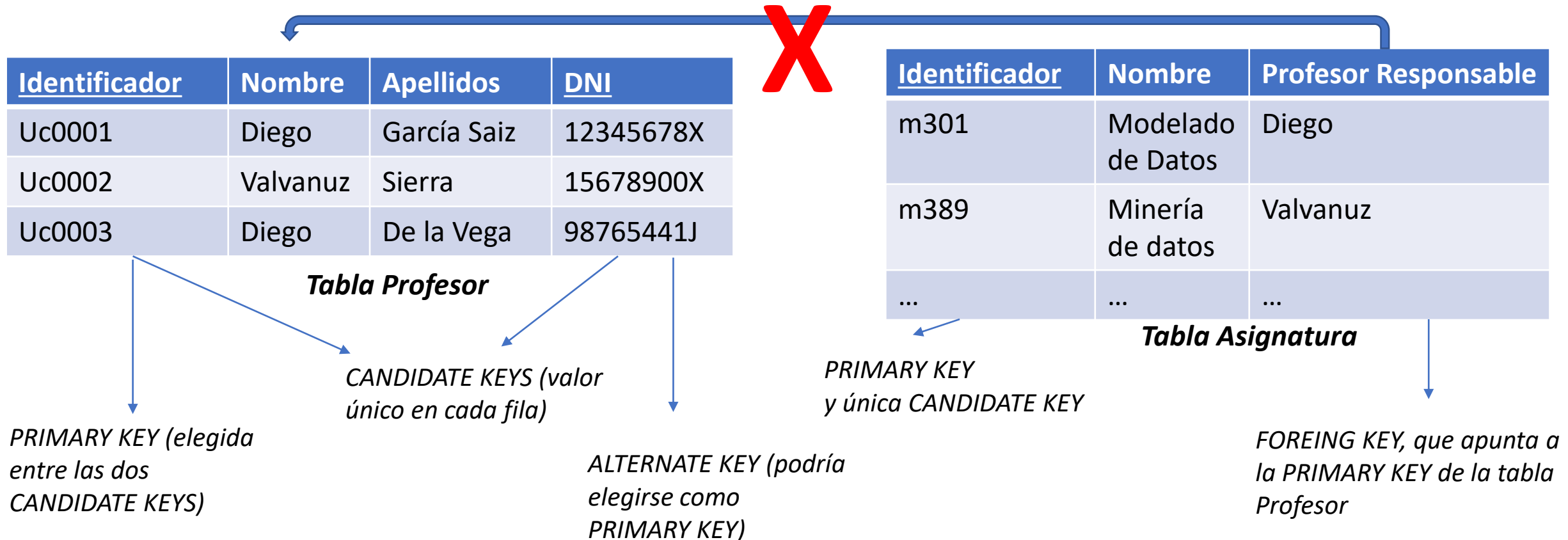
PRIMARY KEY y única CANDIDATE KEY

FOREIGN KEY, que apunta a la PRIMARY KEY de la tabla Profesor

Bases de datos relacionales

- **Atributos, campos o columnas** : PRIMARY KEY (PK), FOREIGN KEY (FK), CANDIDATE KEY (CK) Y ALTERNATE KEY (AK)

UN CAMPO FOREIGN KEY DE UNA TABLA APUNTA A OTRO CAMPO PRIMARY KEY O ALTERNATIVE KEY DE OTRA TABLA. NO PUEDE APUNTAR AL RESTO DE CAMPOS: ESTE DISEÑO ESTA MAL... ¿A QUÉ “DIEGO” SE REFIERE?

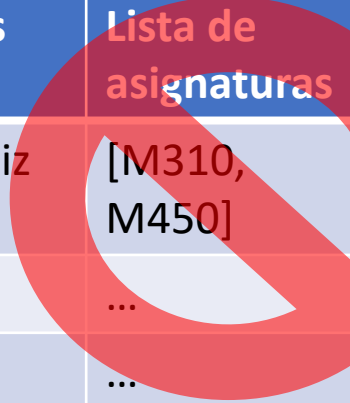


Bases de datos relacionales

- **Atributos, campos o columnas:**

- ¡OJO!: el modelo normalizado de las bases de datos relacionales impide utilizar estructuras de tipo colección, como listas o arrays. Es decir, no se puede crear un campo en la tabla de los profesores que contenga una lista con las asignaturas de las que es responsable.

<u>Identificador</u>	Nombre	Apellidos	Lista de asignaturas
Uc0001	Diego	García Saiz	[M310, M450]
Uc0002	Valvanuz	Sierra	...
...



Bases de datos relacionales

- En una base de datos relacional podemos encontrar los siguientes elementos:
 - **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - Uno a uno (1:1)
 - Uno a varios (1:N)
 - Varios a varios (N:N)

Bases de datos relacionales

- **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - 1:1 -> cada instancia de una entidad se corresponde con una sola instancia de la otra entidad, y viceversa. Se crea un campo en una de las dos entidades para referenciar a las instancias de la otra.
 - Ejemplo: en la Universidad de Cantabria, un plan de estudios tiene un solo profesor responsable, y un profesor sólo puede ser responsable de un plan de estudios.

Opción A.
Referencia
en Plan de
estudios:

Tabla Profesor

<u>Identificador</u>	Nombre	Apellidos	...
Uc0001	Diego	García Saiz	...
Uc0002	Valvanuz	Sierra	...
...

Tabla Plan de estudios

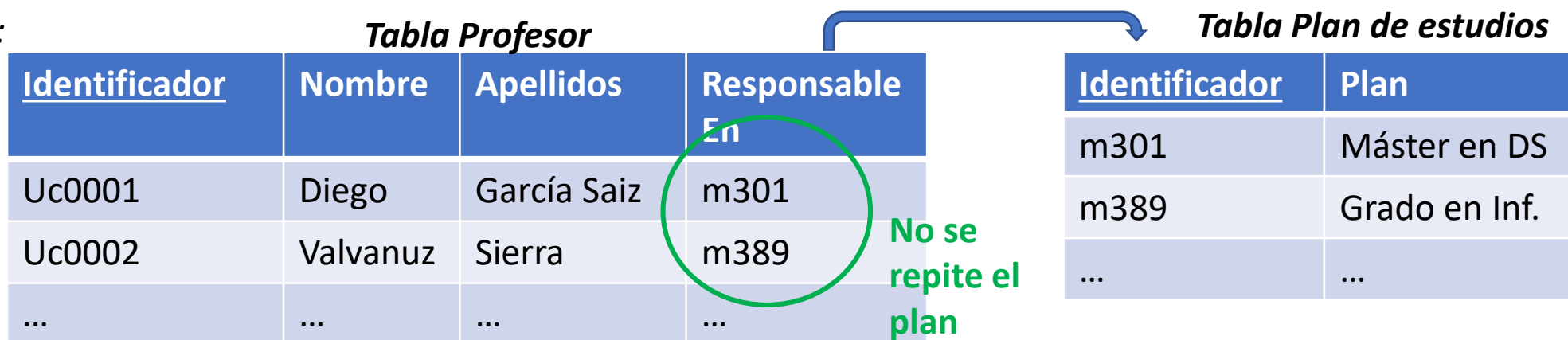
<u>Identificador</u>	Plan	Profesor Responsable
m301	Máster en DS	Uc0001
m389	Grado en Inf.	Uc0002
...

No se repite el
profesor

Bases de datos relacionales

- **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - 1:1 -> cada instancia de una entidad se corresponde con una sola instancia de la otra entidad, y viceversa. Se crea un campo en una de las dos entidades para referenciar a las instancias de la otra.
 - Ejemplo: en la Universidad de Cantabria, un plan de estudios tiene un solo profesor responsable, y un profesor sólo puede ser responsable de un plan de estudios.

Opción B.
Referencia
en Profesor:



Bases de datos relacionales

- **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - 1:N -> cada instancia de una entidad A se corresponde con varias instancias de otra entidad B, pero a cada instancia de B sólo le corresponde una instancia de A. En el modelo relacional, aparece un campo en la entidad B que referencia a una instancia de la entidad A.
 - Ejemplo: las asignaturas sólo tienen un profesor responsable, pero un profesor puede ser responsable en varias asignaturas.

Tabla Profesor

Identificador	Nombre	Apellidos	...
Uc0001	Diego	García Saiz	...
Uc0002	Valvanuz	Sierra	...
...

Tabla Asignatura

Identificador	Nombre	Profesor Responsable
m301	Modelado de Datos	Uc0001
m389	Minería de datos	Uc0001
g768	Programación	Uc0002

Se repite el profesor

Bases de datos relacionales

- **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - N:N -> cada instancia de una entidad A se corresponde con varias instancias de otra entidad B, y a cada instancia de B le pueden también corresponder varias instancias de A. En el modelo relacional, se crea una tabla intermedia que almacena la relación.
 - Ejemplo: las asignaturas tienen matriculados a varios estudiantes, y a su vez los estudiantes están matriculados en varias asignaturas.

Tabla Asignatura

<u>Identificador</u>	Nombre
m301	Modelado de Datos
m389	Minería de datos
g768	Programación

<u>IdAsig</u>	<u>IdEst</u>
m301	Mg01
m389	Mg01
m301	Ma08

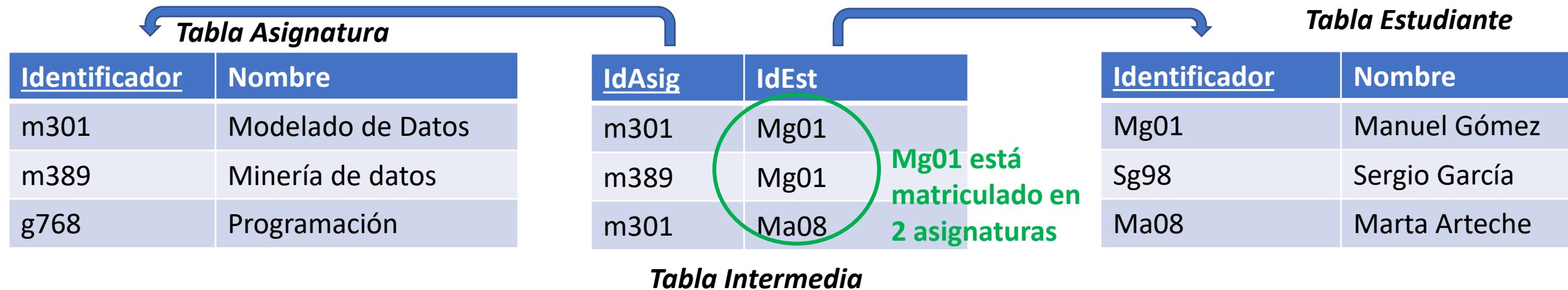
Tabla Intermedia

Tabla Estudiante

<u>Identificador</u>	Nombre
Mg01	Manuel Gómez
Sg98	Sergio García
Ma08	Marta Arteché

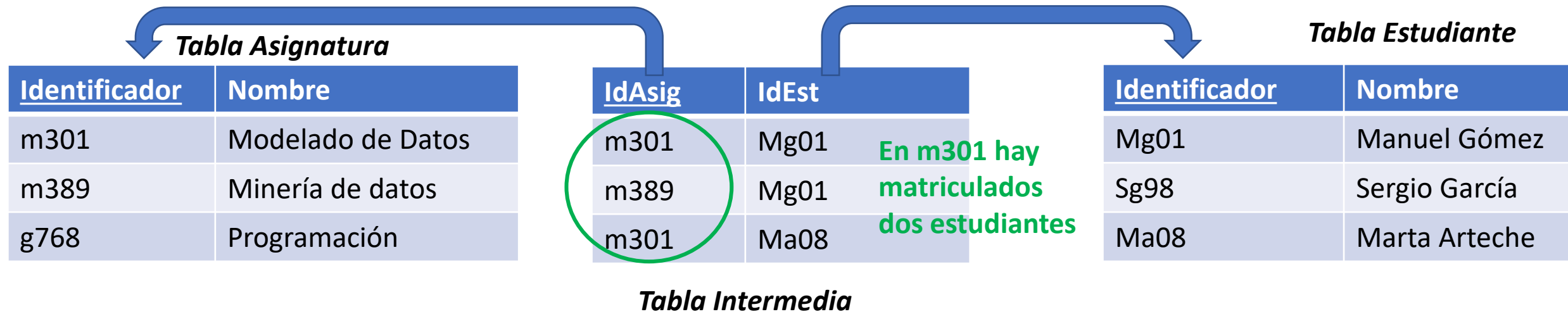
Bases de datos relacionales

- **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - N:N -> cada instancia de una entidad A se corresponde con varias instancias de otra entidad B, y a cada instancia de B le pueden también corresponder varias instancias de A.
 - Ejemplo: las asignaturas tienen matriculados a varios estudiantes, y a su vez los estudiantes están matriculados en varias asignaturas.



Bases de datos relacionales

- **Relaciones entre entidades:** definen como se relacionan las diferentes entidades de la base de datos. Pueden ser de 3 tipos:
 - N:N -> cada instancia de una entidad A se corresponde con varias instancias de otra entidad B, y a cada instancia de B le pueden también corresponder varias instancias de A.
 - Ejemplo: las asignaturas tienen matriculados a varios estudiantes, y a su vez los estudiantes están matriculados en varias asignaturas.



Transacciones ACID

- En una base de datos existen tres tipos de transacciones básicas:
 - Introducir nuevos datos
 - Actualizar los datos ya presentes
 - Eliminar datos
- Las transacciones de los Sistemas Gestores de Bases de Datos Relacionales (en adelante, SGBDR) tienen que soportar ACID:
 - Atomicity (Atomicidad)
 - Consistency (Consistencia)
 - Isolation (Asilamiento)
 - Durability (Durabilidad)

Transacciones ACID

- Atomicity (Atomicidad): cuando se ejecuta una transacción, o se ejecuta de forma completa, o no se ejecuta.
 - Por ejemplo, si queremos sumar 1 crédito a todas las asignaturas del ejemplo anterior, y sucede un error en el sistema después de actualizar la primera asignatura, se cancela por completo la actualización, y se vuelve al estado anterior.
 - Es decir, o se realiza la transacción por completo, o no se realiza.
- Si no hay errores, se actualizan los créditos de todas las asignaturas.

<u>Identificador</u>	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	6
...

Tabla Asignaturas antes del actualizado



Actualizado: suma 1 a todas las asignatura.

<u>Identificador</u>	Nombre	Créditos
m301	Modelado de Datos	7
m389	Minería de datos	7
...

Tabla Asignaturas después del actualizado

Transacciones ACID

- Atomicity (Atomicidad): cuando se ejecuta una transacción, o se ejecuta de forma completa, o no se ejecuta.
 - Por ejemplo, si queremos sumar 1 crédito a todas las asignaturas del ejemplo anterior, y sucede un error en el sistema después de actualizar la primera asignatura, se cancela por completo la actualización, y se vuelve al estado anterior.
 - Es decir, o se realiza la transacción por completo, o no se realiza.

Imaginemos que ha fallado tras actualizar la primera asignatura... ¿se actualiza sólo una fila?: ¡ESTO NO PUEDE SUCEDER!

Identificador	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	6
...

Tabla Asignaturas antes del actualizado

Se produce un error en la actualización



Actualizado: suma 1 a todas las asignatura.

Identificador	Nombre	Créditos
m301	Modelado de Datos	7
m389	Minería de datos	6
...

Tabla Asignaturas después del actualizado

Transacciones ACID

- Atomicity (Atomicidad): cuando se ejecuta una transacción, o se ejecuta de forma completa, o no se ejecuta.
 - Por ejemplo, si queremos sumar 1 crédito a todas las asignaturas del ejemplo anterior, y sucede un error en el sistema después de actualizar la primera asignatura, se cancela por completo la actualización, y se vuelve al estado anterior.
 - Es decir, o se realiza la transacción por completo, o no se realiza.
- No se actualiza NINGUNA asignatura.

Identificador	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	6
...

Tabla Asignaturas antes del actualizado

Se produce un error en la actualización



Actualizado: suma 1 a todas las asignatura.

Identificador	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	6
...

Tabla Asignaturas después del actualizado

Transacciones ACID

- Consistency (consistencia): los datos son consistentes, cumpliéndose las restricciones de integridad definidas.
 - Por ejemplo, no se debe permitir introducir datos de tipo string en campos definidos como numéricos, o no se puede tener dos instancias en una tabla con un valor repetido cuando este deba de ser único (PRIMARY KEY).

<u>Identificador</u>	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	Seis
...

Tabla Asignaturas



Si el campo créditos esta definido como numérico, esto no está permitido.

Transacciones ACID

- Consistency (consistencia): los datos son consistentes, cumpliéndose las restricciones de integridad definidas.
 - Por ejemplo, no se debe permitir introducir datos de tipo string en campos definidos como numéricos, o no se puede tener dos instancias en una tabla con un valor repetido cuando este deba de ser único (PRIMARY KEY).

Si el campo
Identificador es la
PK, no pueden
repetirse los valores



<u>Identificador</u>	Nombre	Créditos
m301	Modelado de Datos	6
m301	Minería de datos	6
...

Tabla Asignaturas

Transacciones ACID

- Consistency (consistencia): los datos son consistentes, cumpliéndose las restricciones de integridad definidas.
 - Por ejemplo, no se debe permitir introducir datos de tipo string en campos definidos como numéricos, o no se puede tener dos instancias en una tabla con un valor repetido cuando este deba de ser único (PRIMARY KEY).

Los valores de la PK no se repiten en diferentes filas

<u>Identificador</u>	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	6
...

En todas las filas los créditos son de tipo numérico

Tabla Asignaturas

Transacciones ACID

- Isolation (aislamiento): si varias aplicaciones acceden de forma concurrente a los datos de una tabla, cada acceso debe realizarse de forma que no afecte a los otros..
 - Es decir, las transacciones de las diferentes aplicaciones han de ejecutarse de forma independiente.
 - El aislamiento puede ser definido en el gestor con distintos niveles. Esto afectará a cómo los cambios que una aplicación realice puedan ser vistos por otra.
 - Por ejemplo, un nivel de aislamiento estricto implicaría que, si una aplicación modifica los créditos de una asignatura, y entra otra aplicación a modificar la misma, esta última debe esperar a que la primera termine.

Transacciones ACID

- Durability (durabilidad): los cambios en la base de datos persisten en el tiempo, incluso si se produce un fallo del sistema.

Identificador	Nombre	Créditos
m301	Modelado de Datos	6
m389	Minería de datos	6
...

Tabla Asignaturas antes del actualizado



Actualizado: suma 1 a todas las asignatura.

Identificador	Nombre	Créditos
m301	Modelado de Datos	7
m389	Minería de datos	7
...

Tabla Asignaturas después del actualizado



Identificador	Nombre	Créditos
m301	Modelado de Datos	7
m389	Minería de datos	7
...

Se recupera el sistema



¡FALLO DEL SISTEMA, LA BASE DE DATOS “SE CAE”!

La Tabla Asignaturas mantiene los cambios de las últimas transacciones confirmadas (persistencia)

Arquitectura de los SGBDR: ANSI-SPARC

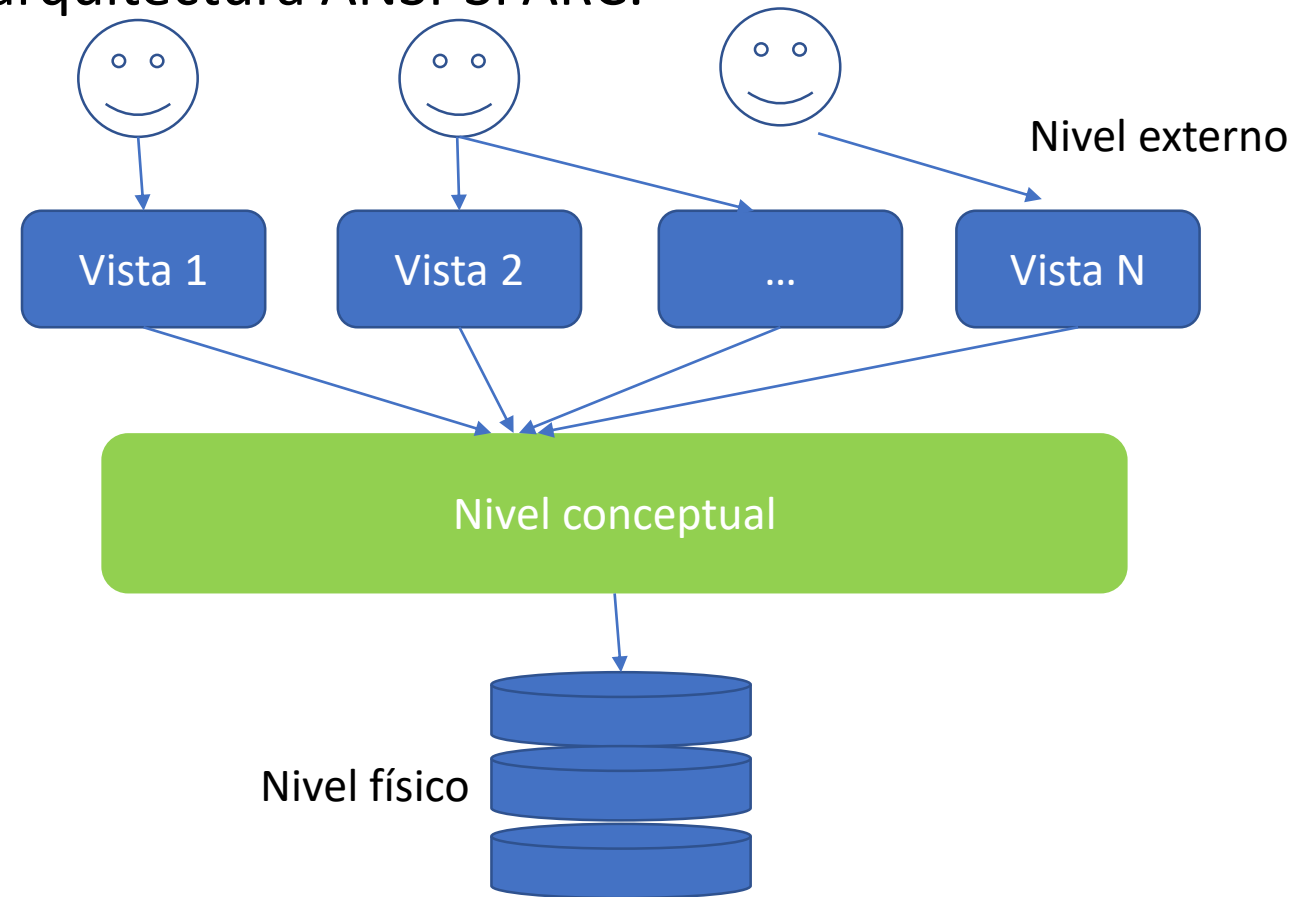
- En los Sistemas Gestores de Bases de Datos Relacionales, existe una independencia lógica y física de los datos. Ejemplo de arquitectura ANSI-SPARC:

- Cada usuario tiene sus propias vistas de acceso independientes. Cambios en las vistas de un usuario no deben afectar a las vistas del resto.

- El sistema de almacenamiento es opaco para los usuarios, no teniendo que conocer su implementación ni resto de detalles.

- El encargado de gestionar la base de datos ha de poder realizar cambios sobre la estructura sin que esto afecte a las vistas de los usuarios

- La estructura de la base de datos no debe verse afectada por cambios físicos, por ejemplo, por un cambio en el medio físico de almacenamiento.

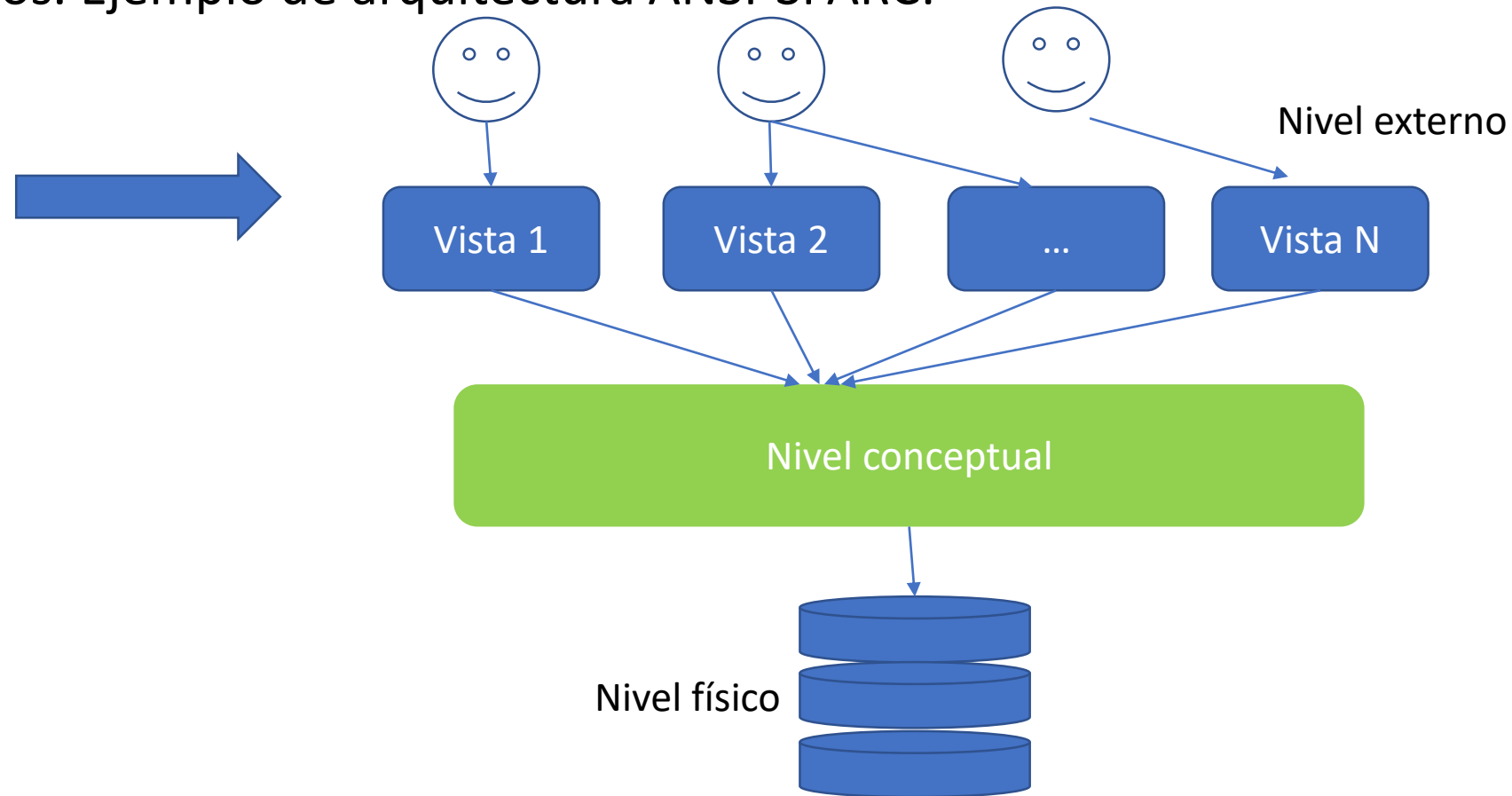


Arquitectura de los SGBDR: ANSI-SPARC

- En los Sistemas Gestores de Bases de Datos Relacionales, existe una independencia lógica y física de los datos. Ejemplo de arquitectura ANSI-SPARC:

- Nivel externo:

- Los usuarios acceden a través de una interfaz o capa externa (vistas).
- No conocen la vicisitudes de la implementación física para el almacenamiento de los datos (ni lo necesitan).
- Además, añade seguridad en los accesos, al poder existir vistas personalizadas para cada tipo de usuario. Por ejemplo:
 - A un profesor de universidad se le puede ofrecer la posibilidad de ver los datos de sus alumnos.
 - A un alumno se le permite ver los datos de las asignaturas en las que se ha matriculado, pero no los datos del resto de alumnos.

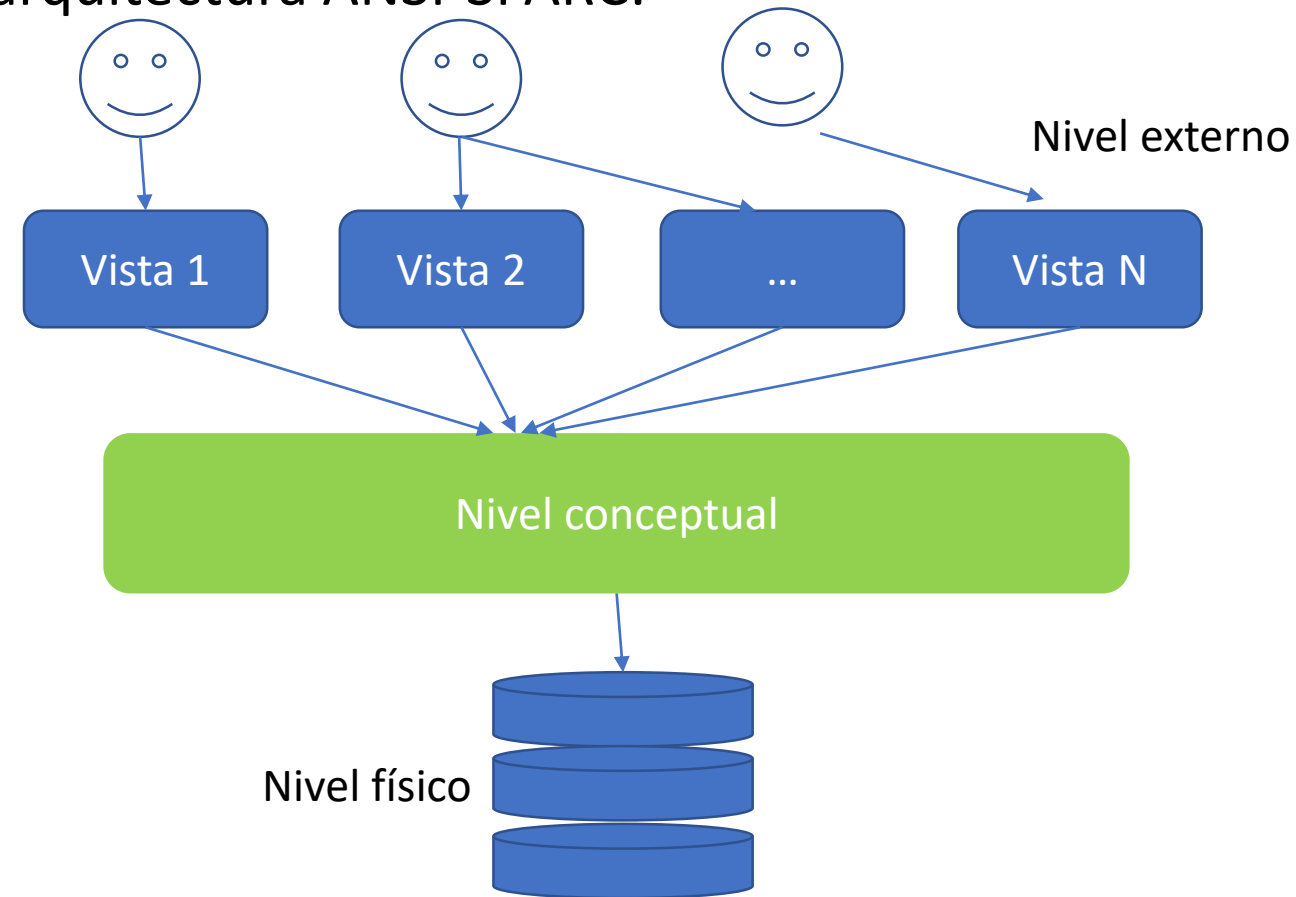


Arquitectura de los SGBDR: ANSI-SPARC

- En los Sistemas Gestores de Bases de Datos Relacionales, existe una independencia lógica y física de los datos. Ejemplo de arquitectura ANSI-SPARC:

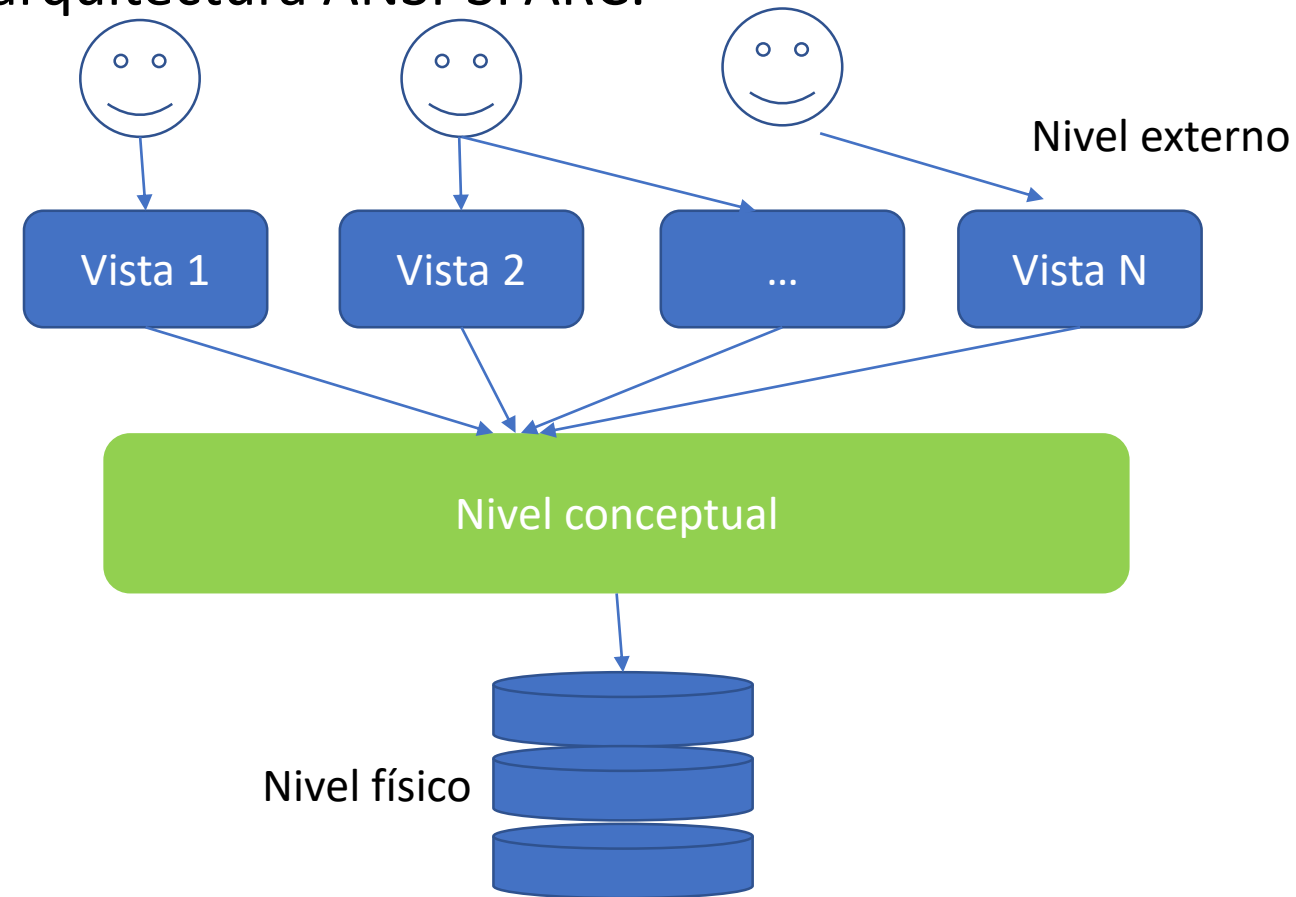
- Nivel Conceptual:

- Describe los datos y sus relacionales. Por ejemplo:
 - Las tablas que existen (profesor, alumno, asignatura...)
 - Los campos de cada tabla (nombre del profesor, créditos de la asignatura)
 - Las relacionales entre las tablas (una asignatura tiene un profesor responsable, un estudiante se matricula de varias asignaturas...)



Arquitectura de los SGBDR: ANSI-SPARC

- En los Sistemas Gestores de Bases de Datos Relacionales, existe una independencia lógica y física de los datos. Ejemplo de arquitectura ANSI-SPARC:



- Nivel Físico:

- Es la forma en la que se representa físicamente la base de datos en un SGBDR concreto.



Sistemas Gestores de Bases de Datos Relacionales

- Transact-SQL (T-SQL)
 - Comercializado por Microsoft.
 - Existen versiones de pago y versiones gratuitas.
- MySQL
 - Propiedad de Oracle.
 - Libre.
- Oracle SQL
 - También de Oracle.
 - De pago.
- SQLite
 - De dominio público y libre.
 - Permite almacenar la base de datos al completo en un fichero del sistema.
 - Simple y con las funcionalidades básicas.
- Y muchos otros: PostgreSQL, MonetDB, IBM DB2...
- Todos se basan en el lenguaje declarativo SQL.



Structured Query Language (SQL)

- Lenguaje para la creación de bases de datos y manipulación y consulta de los datos.
- Es un lenguaje declarativo, “humanamente sencillo” de entender:

Ejemplo de creación de tabla

```
CREATE TABLE Profesor(  
  idprofesor integer PRIMARY KEY,  
  nombre char(20) not null,  
  apellido1 char(20) not null,  
  apellido2 char(20) null,  
  fechaNacimiento date,  
  dni char(9) UNIQUE not null  
);
```

Ejemplo de consulta sobre tabla

```
SELECT * FROM Profesor WHERE idprofesor = 1;
```

Structured Query Language (SQL)

- Hay un estándar, pero ningún SGBDR le sigue al 100%:
 - Diferentes tipos de datos.
 - Diferentes funciones para realizar agregaciones (sumas, cálculo de medias, etc.).
 - Diferentes formas de definir procedimientos.
 - Diferentes formas de definir funciones.
 - Diferentes mecanismos de seguridad.
 - Etc.
- No obstante, es fácil migrar entre unos y otros. A nivel básico, las consultas, transacciones y otras operaciones son idénticas.

¿Por qué usar bases de datos relacionales?

- Los SGBDR son los sistemas de bases de datos más utilizados.
- Tienen numerosas ventajas con respecto a otros sistemas:
 - Modelo normalizado, sencillo de interpretar
 - Garantizan ACID
 - Importante cuando la consistencia y la integridad de los datos es primordial
 - Proveen de un lenguaje sencillo de utilizar para manipular y consultar datos, SQL
 - Es sencillo migrar de un sistema relacional a otro
 - Hay numerosas herramientas que permiten el modelado de datos relacional y su posterior “traducción” a SQL en el sistema SGBDR deseado
 - Los SGBDR son capaces de proveer de capas de abstracción de garanticen la independencia de los modelos conceptuales y físicos, así como de la capa de acceso para los usuarios (vistas)

¿Cuándo no usar bases de datos relacionales?

- El problema de garantizar transacciones ACID es que estas pueden afectar al rendimiento del sistema.
- Por tanto, puede estar recomendado usar otro tipo de BDs cuando la consistencia de los datos no sea fundamental, y haya otros factores más relevantes (ACID no necesario):
 - Necesidad de responder a masivas peticiones de clientes.
 - Manejo de ingentes cantidades de datos para su análisis (Big Data).
 - Los datos no responden al paradigma relacional (geográficos, multimedia, etc).
- Para estos casos, otros paradigmas como NoSQL pueden ser más recomendables.
 - Lo veremos más adelante.