

Relaciones de Ejercicios

Fundamentos de Bases de Datos

Ismael Sallami Moreno

`ism350zsallami@correo.ugr.es`

`https://ismael-sallami.github.io/`

`https://elblogdeismael.github.io/`

Universidad de Granada

Licencia

Este trabajo está licenciado bajo una [Licencia Creative Commons Reconocimiento-NoComercial-SinObraDerivada 4.0 Internacional](https://creativecommons.org/licenses/by-nc-nd/4.0/). <https://creativecommons.org/licenses/by-nc-nd/4.0/>

Usted es libre de:

- Compartir — copiar y redistribuir el material en cualquier medio o formato.

Bajo los siguientes términos:

- **Reconocimiento** — Debe otorgar el crédito adecuado, proporcionar un enlace a la licencia e indicar si se han realizado cambios. Puede hacerlo de cualquier manera razonable, pero no de una manera que sugiera que tiene el apoyo del licenciante o lo recibe por el uso que hace.
- **NoComercial** — No puede utilizar el material para fines comerciales.
- **SinObraDerivada** — Si remezcla, transforma o crea a partir del material, no puede distribuir el material modificado.



Índice general

1. Introducción y definiciones iniciales	5
1.1. Relación de Ejercicios T1	5
2. Arquitectura de un SGBD	9
2.1. Relación de Ejercicios T2	9
3. Modelos de datos	13
3.1. Relaciones de Ejercicios T3	13
3.1.1. Relación A	13
3.1.2. Relación B	18

Capítulo 1

Introducción y definiciones iniciales

1.1. Relación de Ejercicios T1

1. ¿Cuáles son las principales diferencias entre un sistema de procesamiento de archivos y un sistema de bases de datos?

Por un lado, en un sistema de procesamiento de archivos los datos se almacenan en archivos y para trabajar con ellos se utilizan programas específicos. Esto hace que sean desordenados, con repeticiones causando problemas de organización. Además, cabe destacar que la seguridad es limitada.

Por otro lado, en un sistema de bases de datos, los datos se estructuran en tablas relacionadas entre sí. Para gestionar y manipular estos datos, se utiliza un lenguaje de consulta estructurado conocido como SQL, que facilita su manejo de manera eficiente y segura. Además, las bases de datos ofrecen mejores mecanismos de seguridad y son altamente escalables, permitiendo gestionar grandes volúmenes de datos y múltiples usuarios simultáneamente sin inconvenientes.

2. Describe las características más importantes en un sistema de base de datos y también las propiedades más deseables. Explica a tu juicio cuál es la propiedad más importante.

Características y propiedades más importantes de un SGBD:

- a) Estructura y organización: los datos están estructurados de manera eficiente, lo que permite un acceso rápido y fácil a la información.
- b) Integridad: los datos deben ser precisos y consistentes, evitando errores y duplicaciones.
- c) Seguridad: el acceso a los datos debe estar controlado y protegido contra accesos no autorizados.
- d) Escalabilidad: el sistema debe ser capaz de manejar un aumento en la cantidad de datos y usuarios sin perder rendimiento.
- e) Confiabilidad: el sistema debe ser capaz de recuperarse de fallos y mantener la disponibilidad de los datos.

Bajo mi punto de vista, la propiedad más importante es la seguridad ya que de esta manera se evitan accesos a la información no autorizados y otros problemas, como la pérdida de datos o la corrupción de la base de datos. La

seguridad es fundamental para proteger la información sensible y garantizar la confianza de los usuarios en el sistema.

3. Hemos conocido las ventajas de utilizar un sistema de bases de datos, ¿podrías comentar también algunos inconvenientes?

- **Costo:** La implementación y configuración de un sistema de bases de datos puede ser costosa, especialmente en proyectos grandes o en empresas con necesidades complejas. Además, el mantenimiento continuo del sistema también implica costos adicionales.
- **Complejidad:** Es necesario contar con personal capacitado con conocimientos técnicos especializados para gestionar y utilizar la base de datos de manera eficiente. Asimismo, la organización y estructuración de grandes volúmenes de datos o relaciones complejas entre entidades puede ser un desafío.
- **Seguridad:** Las bases de datos pueden ser vulnerables a ataques de hackers interesados en acceder a información sensible. Por ello, es fundamental implementar medidas de seguridad sólidas y contar con un plan de recuperación ante desastres, lo que puede aumentar la complejidad de la gestión.

4. Explica la diferencia entre independencia física e independencia lógica.

La independencia física permite cambiar cómo se almacenan los datos sin afectar su uso lógico, es decir, que si se cambia el hardware las aplicaciones no se ven afectadas, mientras que la independencia lógica permite modificar la organización y presentación de los datos sin alterar su almacenamiento físico. En esencia, la primera se centra en el almacenamiento y la segunda en la estructura lógica de cara al usuario.

5. Definir brevemente los siguientes conceptos:

- **Base de datos:** Conjunto de datos relacionados y organizados de manera estructurada, que se almacenan y gestionan mediante un sistema de gestión de bases de datos (SGBD).
- **DBMS (DataBase Management System):** Programas para describir las estructuras y gestionar la información de BD.
- **DBA (DataBase Administrator):** Persona responsable de la administración y gestión de una base de datos, asegurando su rendimiento, seguridad y disponibilidad.
- **Acceso concurrente:** Capacidad de múltiples usuarios para acceder y manipular datos en una base de datos al mismo tiempo, garantizando la integridad y consistencia de la información.
- **Vista de usuario:** Representación personalizada de los datos en una base de datos, que permite a los usuarios interactuar con la información de manera específica y adaptada a sus necesidades, sin necesidad de conocer la estructura interna de la base de datos.

6. Explicar brevemente los conceptos de: Integridad, fiabilidad y seguridad en una base de datos.

- Ordenarlos por importancia, explicando los criterios utilizados para elaborar el orden.
 - a) Seguridad: Es fundamental proteger la información sensible y garantizar que solo los usuarios autorizados tengan acceso a los datos. Sin seguridad, la integridad y fiabilidad de los datos pueden verse comprometidas.
 - b) Integridad: La precisión y consistencia de los datos son esenciales para garantizar que la información sea útil y confiable. Si los datos no son íntegros, las decisiones basadas en ellos pueden ser erróneas.
 - c) Fiabilidad: Aunque la fiabilidad es importante, si los datos son seguros e íntegros, la fiabilidad se convierte en un aspecto secundario. La fiabilidad se basa en la seguridad y la integridad de los datos, por lo que su importancia es menor en comparación con los otros dos aspectos.
- ¿En qué etapa de la vida de una base de datos se deben tener en cuenta unos y otros?
 - Seguridad: Desde el inicio del diseño de la base de datos, es fundamental establecer políticas y medidas de seguridad para proteger la información.
 - Integridad: Durante la fase de diseño y modelado de la base de datos, se deben definir reglas y restricciones para garantizar la integridad de los datos.
 - Fiabilidad: A lo largo de toda la vida útil de la base de datos, es importante mantener un sistema confiable mediante copias de seguridad y recuperación ante desastres.
- ¿Cómo se mantienen en una base de datos?
 - Seguridad: Se implementan controles de acceso, autenticación y autorización, así como cifrado de datos y auditorías de seguridad.
 - Integridad: Se utilizan restricciones de integridad, como claves primarias, foráneas y reglas de validación, para asegurar la calidad de los datos.
 - Fiabilidad: Se realizan copias de seguridad periódicas, pruebas de recuperación y mantenimiento preventivo del sistema para garantizar su disponibilidad y rendimiento.

Capítulo 2

Arquitectura de un SGBD

2.1. Relación de Ejercicios T2

1. Explicar la relación existente entre los niveles de una base de datos y el concepto de independencia.

La relación entre los niveles de una base de datos y el concepto de independencia radica en la estructura de tres niveles propuesta por el modelo ANSI/SPARC, que busca separar la forma en que los datos se almacenan, representan y acceden. Estos niveles son:

- **Nivel interno (físico):** Describe cómo se almacenan los datos físicamente en el sistema de almacenamiento.
- **Nivel conceptual:** Proporciona una vista abstracta de la base de datos, definiendo la estructura de los datos y sus relaciones sin preocuparse por los detalles de almacenamiento.
- **Nivel externo (vista de usuario):** Presenta a los usuarios una versión personalizada de los datos, mostrando solo la información relevante para ellos.

Independencia de los datos

Este concepto se divide en dos tipos:

- **Independencia lógica de los datos:** Se refiere a la capacidad de modificar el esquema conceptual sin afectar los esquemas externos. Por ejemplo, se pueden agregar nuevas entidades o atributos sin necesidad de cambiar la forma en que los usuarios ven los datos.
- **Independencia física de los datos:** Permite cambiar la forma en que los datos están almacenados (como cambiar de disco duro o modificar estructuras de almacenamiento) sin afectar la forma en que los datos son representados en los niveles superiores.

Podemos concluir con que la arquitectura de tres niveles facilita la independencia de datos, permitiendo modificar estructuras internas o lógicas sin afectar la interacción del usuario con la base de datos.

2. Explicar la diferencia entre esquema externo y aplicaciones de usuario.

La diferencia entre el esquema externo y las aplicaciones de usuario radica en su propósito y función dentro del sistema de bases de datos.

Esquema Externo

El esquema externo representa la vista de los datos desde la perspectiva de los usuarios. Cada usuario o grupo de usuarios puede tener un esquema externo diferente, dependiendo de sus necesidades y permisos. Se define a partir del esquema conceptual y no afecta la estructura interna de la base de datos.

Por ejemplo, un *cajero* en un banco solo puede ver los saldos y transacciones de clientes, mientras que un gerente puede acceder a reportes financieros más detallados.

Aplicaciones de Usuario

Las aplicaciones de usuario son los programas o interfaces que interactúan con la base de datos para realizar operaciones como consultas, inserciones, modificaciones o eliminaciones de datos. Estas aplicaciones pueden utilizar SQL, interfaces gráficas o APIs para comunicarse con la base de datos y presentar los datos según el esquema externo definido para el usuario.

Diferencia Clave

- El esquema externo es una estructura lógica que define qué datos puede ver un usuario y cómo están organizados.
- Las aplicaciones de usuario son herramientas o programas que permiten interactuar con la base de datos utilizando el esquema externo para presentar la información.

El esquema externo define la visión restringida de los datos para cada usuario, mientras que las aplicaciones de usuario son los medios mediante los cuales se accede y manipulan esos datos.

3. Explica el motivo por el que, a tu juicio, no se han desarrollado DDLs a nivel interno.

No se han desarrollado DDLs a nivel interno porque la gestión del almacenamiento físico *es responsabilidad del SGBD*. Esto garantiza eficiencia, independencia de datos, seguridad y portabilidad, evitando que los usuarios interfieran con la optimización del sistema.

4. Explica el motivo por el que, a tu juicio, no se han desarrollado DMLs a nivel externo.

No se han desarrollado DDLs a nivel interno porque el SGBD gestiona automáticamente el almacenamiento físico para optimizar rendimiento, garantizar independencia de datos y evitar configuraciones ineficientes por parte del usuario.

5. Buscar tres ejemplos de lenguajes de cuarta generación. Indicar sus objetivos o funciones.

6. ¿Cuál es el enfoque actual del concepto de lenguaje anfitrión? Dar ejemplos de lenguajes anfitrión.

El lenguaje anfitrión es un lenguaje de programación que integra comandos SQL para interactuar con una base de datos. Actualmente, el enfoque se centra en la integración fluida con los SGBD y la seguridad en la ejecución de consultas.

Ejemplos: Java (JDBC), Python (SQLAlchemy), PHP (PDO), C# (Entity Framework).

7. ¿Qué elementos conciernen al nivel interno de una base de datos?

Incluye estructuras físicas de almacenamiento, organización de archivos, índices, métodos de acceso y estrategias de optimización para el almacenamiento y recuperación de datos.

8. ¿Qué cuestiones debe cubrir a tu juicio una buena herramienta de gestión privilegios de usuarios?

- a) Control de acceso (roles y permisos).
- b) Control y registro de actividad.
- c) Administración centralizada.
- d) Compatibilidad con autenticación externa.
- e) Facilidad de uso y escalabilidad.

9. Explicar las ventajas de la arquitectura cliente/servidor a tres niveles.

- a) Escalabilidad: Mejora el rendimiento al distribuir la carga.
- b) Seguridad: Separa la lógica de negocio y el acceso a datos.
- c) Mantenimiento fácil: Cambios en una capa no afectan las demás.
- d) Optimización de recursos: Reduce la dependencia del cliente en el procesamiento.

Capítulo 3

Modelos de datos

3.1. Relaciones de Ejercicios T3

3.1.1. Relación A

1. ¿Cuándo aparece el concepto de modelo de datos? ¿Por qué?

El concepto de modelo de datos surge en la década de 1970 debido a la necesidad de estructurar la información en bases de datos de manera más eficiente y organizada. Antes de esto, los sistemas de gestión de datos utilizaban enfoques desordenados y dependientes de estructuras jerárquicas o de redes, lo que dificultaba la manipulación, recuperación y mantenimiento de la información. Con la llegada del modelo relacional de Edgar F. Codd, se estableció una base teórica sólida para la gestión de datos, promoviendo su independencia del hardware y los programas específicos.

2. Diferencias entre los modelos de datos semánticos y los modelos de datos implementables.

Modelos de datos semánticos: Representan la información a un nivel conceptual, describiendo cómo los datos se relacionan entre sí en el mundo real. Se enfocan en el significado de los datos y en su estructura lógica, sin preocuparse por su almacenamiento físico. Un ejemplo es el Modelo Entidad-Relación (E-R).

Modelos de datos implementables: Son aquellos diseñados para su almacenamiento y gestión en un Sistema de Gestión de Bases de Datos (SGBD). Se derivan de los modelos semánticos y adaptan los datos a estructuras que pueden ser manejadas por un sistema informático, como tablas en el modelo relacional.

3. Define entidad y proporciona algún ejemplo. Distinguir entre conjunto de entidades y entidad concreta.

Entidad: Es cualquier objeto del mundo real que puede identificarse de manera única y que posee atributos que lo describen. Puede ser una persona, un lugar, un evento o un concepto.

Ejemplo: Un Estudiante.

Conjunto de entidades: Representa el grupo de todas las entidades del mismo tipo. En este caso, todos los estudiantes de una universidad.

Entidad concreta: Se refiere a una instancia específica dentro del conjunto de entidades. Ejemplo: Juan Pérez, que es un estudiante concreto dentro del grupo de estudiantes.

4. Para el ejemplo de entidad antes definida, define sus atributos y proporciona un ejemplo. Identificar la clave primaria de ésta.

Atributos del Estudiante: Nombre, Apellido, Matrícula, Fecha de nacimiento, Carrera.

Ejemplo de instancia:

Nombre: Juan

Apellido: Pérez

Matrícula: 12345

Fecha de nacimiento: 01/01/2000

Carrera: Ingeniería Informática

Clave primaria: Matrícula. Este atributo es único para cada estudiante y permite identificarlo sin ambigüedad.

5. Poner un ejemplo de atributo compuesto relacionado con la entidad definida antes.

Un atributo compuesto es aquel que puede dividirse en subcomponentes con significado propio.

Para la entidad Estudiante, un atributo compuesto podría ser la Dirección, que se divide en:

- Calle
- Número
- Ciudad
- Código postal

Ejemplo: “Av. Siempre Viva, 742, Huéscar, 1234”.

6. Definir relación y dar un ejemplo que involucre a la entidad antes definida. Una relación es la asociación entre dos o más entidades. Representa la forma en que los objetos del mundo real están conectados.

Ejemplo: La entidad *Estudiante* puede estar relacionada con la entidad *Curso* mediante la relación *Inscrito_en*, indicando en qué cursos está matriculado cada estudiante.

7. Dar un ejemplo de relación de grado mayor que dos, referente a la entidad antes definida. Una relación de grado mayor que dos involucra tres o más entidades.

Ejemplo: La relación *Evalúa* entre las entidades *Estudiante*, *Profesor* y *Curso*, donde un profesor evalúa a un estudiante en un curso específico.

8. Definir cardinalidad máxima y mínima y dar dos ejemplos: uno referente a una relación uno a muchos y otro referente a una relación muchos a muchos. La cardinalidad mínima indica el número mínimo de veces que una entidad puede participar en una relación. La cardinalidad máxima define el número máximo de veces que una entidad puede estar en una relación.

Ejemplos:

- Relación uno a muchos:
 - Un *Profesor* puede impartir muchos *Cursos*, pero cada curso solo tiene un profesor.
 - Cardinalidad: (*Profesor* 1:N *Curso*).
 - Relación muchos a muchos:
 - Un *Estudiante* puede inscribirse en varios *Cursos* y un curso puede tener varios estudiantes.
 - Cardinalidad: (*Estudiante* M:N *Curso*).
9. Definir entidad débil, dar dos ejemplos diferentes de los estudiados en clase. Una entidad débil es aquella que no puede identificarse de manera única sin la ayuda de otra entidad, llamada entidad fuerte, de la que depende.
- *Historial_Médico*, que depende de la entidad fuerte *Paciente*.
 - *Dependiente*, que depende de la entidad fuerte *Empleado* en una empresa.
10. Distinguir mediante ejemplos los conceptos de entidad débil y de relación uno a muchos.
- Entidad débil:
 - Un *Dependiente* no puede existir sin un *Empleado*, ya que su identificación depende de este.
 - Relación uno a muchos:
 - Un *Profesor* puede impartir varios *Cursos*, pero un curso solo tiene un profesor. Aquí, ambas entidades son independientes, a diferencia de la entidad débil.
11. Establecer la conexión entre el concepto de entidad débil y el de atributo compuesto multivaluado. Ilustrar esta idea mediante dos ejemplos.

Una entidad débil puede representar información que en algunos casos podría modelarse como un atributo compuesto multivaluado, dependiendo del nivel de detalle requerido.

- Entidad débil: Un *Historial_Médico* es una entidad que almacena múltiples registros de un *Paciente*.
 - Alternativa: Un atributo compuesto multivaluado *Historial_Médico* en la entidad *Paciente*, con los campos *Fecha*, *Diagnóstico*, *Tratamiento*.

- Entidad débil: Un *Dependiente* de un *Empleado* (hijo, cónyuge).
 - Alternativa: Un atributo compuesto multivaluado *Dependientes* en la entidad *Empleado* con los campos *Nombre*, *Relación*, *Fecha_Nacimiento*.
- 12. Define el concepto de subtipo, poner dos ejemplos que no se hayan explicado en clase. Un subtipo es una especialización de una entidad que hereda sus atributos y puede agregar otros específicos.
 - *Vehículo* con los subtipos *Coche* y *Motocicleta*.
 - *Dispositivo_Electrónico* con los subtipos *Smartphone* y *Tablet*.

13. Define el concepto de herencia. ¿Cómo se relaciona con el caso anterior?

La herencia en bases de datos es el mecanismo mediante el cual un subtipo hereda atributos y relaciones de una entidad general.

Se relaciona con los subtipos porque estos aplican herencia para compartir atributos comunes. Por ejemplo, *Vehículo* tiene un atributo *Matrícula*, que es heredado por *Coche* y *Motocicleta*.

14. Explicar la diferencia entre la relación TIENE-UN y la relación ES-UN, y poner un ejemplo de cada una de ellas.

La relación **TIENE-UN** indica que una entidad posee otra entidad o atributo, pero ambas siguen siendo entidades distintas. En cambio, la relación **ES-UN** representa una jerarquía de especialización donde una entidad es un subtipo de otra y hereda sus atributos.

- Ejemplo de **TIENE-UN**: Una entidad *Coche* tiene un *Motor*. Aquí, *Motor* es una entidad independiente que se relaciona con *Coche*.
 - Ejemplo de **ES-UN**: Una entidad *Empleado* puede especializarse en *Ingeniero* y *Administrativo*. En este caso, *Ingeniero* y *Administrativo* heredan atributos de *Empleado*.
15. Poner un ejemplo mediante un diagrama entidad/relación de cada una de las siguientes situaciones:
- a) Relación con atributos uno a muchos.
 - b) Relación involutiva muchos a muchos.
 - c) Una relación ternaria con algún atributo adicional.
 - d) Una relación ES-UN donde las subclases tengan atributos adicionales.
 - e) Entidades fuerte y débil donde esta última se conecte con otras entidades del diagrama.
16. Decidir de forma razonada cuándo se debe incluir una entidad nueva junto con una relación, o un atributo en cada uno de los siguientes casos.
- a) La fecha de edición de un libro. **Atributo**, ya que es una propiedad intrínseca del libro.

- b) Los autores de un libro. **Entidad con relación**, porque un autor puede escribir varios libros y un libro puede tener varios autores (relación muchos a muchos).
 - c) El título de una película. **Atributo**, ya que es una característica única de la película.
 - d) Los actores de una película. **Entidad con relación**, ya que un actor puede participar en varias películas y una película puede tener varios actores.
 - e) La valoración psicológica de un alumno. **Entidad con relación**, porque puede haber múltiples valoraciones en diferentes fechas, asociadas a cada alumno.
 - f) La descripción de un accidente. **Atributo**, ya que es una información específica de la entidad **Accidente**.
 - g) Los cuartos de baño que tiene un apartamento. **Atributo**, porque es una propiedad fija del apartamento.
 - h) El programa de una asignatura. **Entidad con relación**, porque el programa puede cambiar con el tiempo y necesita un historial.
 - i) El responsable de una investigación criminal. **Entidad con relación**, ya que un investigador puede encargarse de varias investigaciones.
 - j) La fecha de un crimen. **Atributo**, ya que es una característica propia de la entidad **Crimen**.
 - k) Los datos laborales de los trabajadores. **Entidad con relación**, porque pueden cambiar con el tiempo (salario, cargo, antigüedad, etc.).
 - l) El horario de una asignatura. **Entidad con relación**, ya que una asignatura puede tener múltiples horarios en distintos semestres.
17. Poner un ejemplo mediante el diagrama entidad/relación de cada una de las siguientes situaciones y pasar a tablas el resultado, indicando la clave primaria de la tabla resultante.
- a) Una relación ternaria con algún atributo adicional.
 - b) Una relación ES-UN, donde las subclases tengan atributos adicionales.
 - c) Entidades subordinante y subordinada.
 - d) Tipo y subtipo con herencia de propiedades.
 - e) Relación con atributos uno a muchos.
 - f) Relación involutiva muchos a muchos.

(Nota: Se requiere un diagrama E/R para representar gráficamente estas situaciones y su transformación en tablas relacionales.)

18. Poner ejemplos de los distintos casos que aparecen cuando se pasa de una conexión en el modelo E/R a una tabla en el modelo relacional. Estudiar las distintas posibilidades de clave primaria que pueden aparecer. Los casos principales son:

- Relación uno a uno: La clave primaria de una entidad puede actuar como clave foránea en la otra.
- Relación uno a muchos: La clave primaria de la entidad del lado "uno" se usa como clave foránea en la entidad del lado "muchos".
- Relación muchos a muchos: Se crea una tabla intermedia con claves foráneas de ambas entidades como clave primaria compuesta.

19. ¿Por qué crees que los modelos basados en grafos no mantienen realmente la independencia física?

Porque la estructura del almacenamiento en un modelo basado en grafos depende directamente de la forma en que se almacenan los nodos y las relaciones, lo que afecta el rendimiento según la implementación en la base de datos subyacente.

20. ¿Qué crees que significa la afirmación de que el modelo relacional es simétrico con respecto a las relaciones muchos a muchos?

Significa que las relaciones muchos a muchos en el modelo relacional se representan de manera uniforme mediante una tabla intermedia, sin importar el tipo de entidades involucradas. Esto garantiza una representación consistente de las relaciones complejas.

21. Poner tres ejemplos de conexiones en el modelo E/R que no se trasladen bien al modelo relacional.

- Jerarquías de herencia: Requieren estrategias como el uso de una tabla por subtipo o una tabla con un campo que indique el tipo.
- Relaciones recursivas: Se necesita un mecanismo especial para manejar relaciones en las que una entidad se asocia consigo misma.
- Atributos multivaluados: Deben transformarse en una tabla separada en el modelo relacional.

3.1.2. Relación B

Bibliografía

- [1] Ismael Sallami Moreno, **Estudiante del Doble Grado en Ingeniería Informática + ADE**, Universidad de Granada, 2025.