

## Lectura: Introducción a la gestión de proyectos

Objetivo general de la gestión de proyectos:

- Aprender técnicas y conocimientos para administrar proyectos eficientemente.
- Optimizar recursos aplicando mejores prácticas del Project Management Institute (PMI).

Conceptos clave:

- Proyecto: Esfuerzo temporal para crear un producto, servicio o resultado único.
- Administración de proyectos: Aplicación de conocimientos, habilidades, herramientas y técnicas para cumplir los requerimientos de un proyecto.
- PMI (Project Management Institute): Organización sin fines de lucro que promueve la gestión de proyectos y establece estándares globales.
- PMBOK (Project Management Body of Knowledge): Marco de referencia que describe el entorno, ciclo de vida y conocimientos necesarios para administrar proyectos.

Factores de éxito y fracaso en proyectos:

- Éxito: solo un pequeño porcentaje de proyectos (16.2%) según el CHQS Report).  
Lograr cumplir objetivos en tiempo, costo y calidad.
- Fracaso:
  - Expectativas poco realistas.
  - Requerimientos cambiantes o incompletos.
  - Falta de involucramiento de usuarios.
  - Poca preparación técnica del equipo.
  - Uso inadecuado de herramientas y métodos.
  - Falta de soporte señalial y mala planificación.

Áreas clave en la gestión de proyectos (según el PMBOK)

- Integración: Coordinar los distintos elementos del proyecto.
- Alcance: Definir y controlar el trabajo necesario.
- Calendario y costo: Planificar la secuencia de actividades y asignar recursos.
- Calidad: Satisfacer los objetivos mediante mejores continuas.
- Riesgo: Identificar y gestionar eventos que puedan afectar el producto.
- Recursos Humanos: Gestionar el equipo de trabajo y su participación.
- Recursos Materiales: Aprovisionamiento y selección de proveedores.
- Comunicación: Manejo adecuado de la información del proyecto.

Preguntas: Según su experiencia, ¿cuál de las áreas de la gestión de proyectos suele ser la más problemática en la práctica, y cómo se puede mejorar su manejo?

## Construcción de software

### Sistemas de Información en los Negocios

Introducción a los sistemas de información

- La tecnología ha transformado la administración, finanzas y producción, facilitando tareas mecánicas y rutinarias.
- La digitalización ha permitido el acceso a grandes volúmenes de información mediante bases de datos en línea e Internet.

Impacto en los negocios

- Los sistemas de información son clave en la estrategia empresarial, generando ventajas competitivas.
- El director de informática ahora es un gerente estratégico, no solo un técnico.
- La correcta elección de tecnologías permite mejorar la toma de decisiones y la eficiencia organizacional.

Definición de un Sistema de Información (SI)

concepto técnico (Laudon & Laudon, 2004):

- Conjunto de componentes interrelacionados que recopilan, procesan, almacenan y distribuyen información para la toma de decisiones y el control organizacional.

Elementos del SI:

- Entrada (captura de datos).
- Procesamiento (clasificación, cálculo, validación).
- Salida (información útil para el usuario).
- Retroalimentación (mejora continua del sistema).

TIPOS de sistemas de información

1. TPS (Transaction Processing System)

- Manejan las transacciones diarias de la empresa.
- Ejemplos: sistemas de facturación, procesamiento de ventas.
- Características: gran volumen de datos, alta repetición, requiere almacenamiento fijo.

2. OAS (Office Automation System)

- Automatizan procesos administrativos y mejoran la productividad.
- Ejemplos: correo electrónico, procesadores de texto, agendas electrónicas.

3. KWS (Knowledge Work Systems)

- Automatizan procesos administrativos y mejoran la productividad.
- Ejemplos: correo electrónico, procesadores de texto, grandes proyectos de software, diseño, modelado 3D.

4. MIS (Management Information Systems)

- Procesan información a seriales para la toma de decisiones organizacionales.
- Tipos de reportes:
  - Programados: se generan periódicamente.
  - Por demanda: se solicitan cuando se requieren.
  - Por excepción: se generan ante situaciones críticas.

5. DSS (Decision Support System)

- Sistemas que ayudan en decisiones semi-estructuradas o no estructuradas.
- Características: manejo de grandes volúmenes de datos, capacidad de análisis complejo, what-if.

## Construcción de Software

### Sistema de Información en los Negocios

#### G-CDSs (Group Decision Support Systems)

- Apoyan la toma de decisiones en grupo con herramientas colaborativas.
- Características: Entrada anónima, flexibilidad en estilos de decisión, facilidad de uso.

#### E-EIS (Expert Systems)

- Sistemas que imitan el funcionamiento de un experto en un área específica.
- Ejemplos: Diagnóstico médico, predicción de eventos financieros.

#### B-EIS (Executive Information Systems)

- Proveen información crítica a altos ejecutivos con indicadores clave y visualizaciones gráficas.

Pregunta: ¿Cómo han evolucionado los sistemas de información en la toma de decisiones estratégicas de las empresas en los últimos años, y cuál cree que será la siguiente gran tendencia en este campo?

### Gestión del alcance

#### Definición y gestión del alcance

- El alcance del proyecto es la suma de productos y servicios que deben entregar.
- Define lo que está y no está incluido en el proyecto.
- Se requiere que los objetivos sean SMART.
- La gestión del alcance asegura que solo se realicen los trabajos necesarios para el éxito del proyecto.

#### Diferencia entre alcance del producto y del proyecto

- Alcance del producto: características y funciones del producto o servicio.
- Alcance del proyecto: Trabajos o tareas necesarios para entregar el producto con las especificaciones requeridas.

#### Procesos de gestión del alcance

1. Iniciación y autorización formal del proyecto, se definen objetivos generales y se asigna un jefe de proyecto.
2. Planificación del alcance: creación de un conjunto detallado del alcance (WBS, fases).
3. Definición del alcance: se descompone el trabajo en partes más manejables, facilitando su control.
4. Verificación del alcance: Evaluación de entregables para asegurar que cumplen los requisitos.
5. Control de cambios del alcance: supervisión de modificaciones para evitar scope creep (desviaciones no controladas).

#### Herramientas clave: Work Breakdown Structure (WBS)

- Descomposición jerárquica de las tareas del proyecto en elementos manejables.
- Cada elemento debe representar un entregarible medible y único.
- Permite estimar tiempos, costos y recursos con precisión.
- Se debe descomponer hasta un nivel que permita el control efectivo sin sobrecargar el seguimiento.

## CONSTRUCCIÓN DE SOFTWARE

### VERIFICACIÓN Y CONTROL DEL ALIANCE

- Se requiere la verificación formal de los entregables por parte del cliente o patrocinador.
- Se realizan inspecciones y auditorías para asegurar la calidad y confiabilidad.
- Se documentan cambios y se tratan a través de un sistema de control formal.

Presentación:

¿Cuáles son las mejores prácticas para editar el scope creep en un proyecto y cómo se presenta ésta en entornos con múltiples interesados con expectativas diferentes?

### BD VS DBMS

Definición y contexto de bases de datos

Antes de la existencia de las bases de datos, los datos estaban ligados a los programas de aplicación, lo que generaba redundancia, inconsistencias y pérdida de accesibilidad en la información. Con el anfuge de bases de datos, surgida en los años 60, se buscó diseñar un sistema centralizado donde los datos se estructuraran de manera eficiente para evitar duplicidades y garantizar coherencia.

¿Qué es una base de datos?

Una base de datos es una colección de archivos interrelacionados que minimiza la repetición de datos y organiza la información para cubrir las necesidades de una entidad. Su diseño debe garantizar la integridad y eficiencia en el acceso a los datos.

¿Cuando conviene usar base de datos?

Las bases de datos son esenciales cuando se manejan múltiples archivos interrelacionados o grandes volúmenes de datos, como en empresas de manufactura o registros de gran escala (elecciones, directorios telefónicos, etc.).

### Sistemas de gestión de bases de datos (SGBD)

Un DBMS es un sistema que permite gestionar bases de datos de manera eficiente y segura. Su objetivo es facilitar el acceso, garantizar la integridad y proteger la información contra fallos y accesos no autorizados.

Problemas en sistemas tradicionales de almacenamiento

Los sistemas basados en archivos presentan varias limitaciones, como:

- Redundancia e inconsistencia: datos duplicados y posibles errores al actualizar la información.
- Dificultad de acceso: consultas no provistas requieren nuevos programas.
- Rastreo de datos: dificultad para integrar información de distintos archivos.
- Problemas de concurrencia: conflictos en actualizaciones simultáneas.
- Seguridad e integridad: dificultad para restringir usuarios y garantizar reglas de negocio.

### Administrador de bases de datos (DBMS)

El DBMS actúa como intermediario entre los datos almacenados y los programas de aplicaciones. Sus principales funciones incluyen:

- Interacción con el sistema de archivos para almacenar y recuperar datos.
- Garantizar la integridad y seguridad de la información.
- Reglas socias de seguridad y recuperación ante fallos del sistema.
- Control de concurrencia para evitar inconsistencias en actualizaciones simultáneas.

## Construcción de Software

### BD vs DBMS

Presentes, éstos son los principales criterios para elegir un DBMS adecuado según el tipo y volumen de datos de una empresa.

### Navegación del módulo Entidad-Relación y restricciones adicionales

El diseño de bases de datos relacionales buscan minimizar la redundancia y facilitar la recuperación de información. Para ello, se utilizan esquemas normalizados que dependen de las restricciones de los datos. Un mal diseño puede generar redundancia, inconsistencias y desperdicio de espacio de almacenamiento.

#### Fases del diseño de bases de datos

1. Recolección y análisis de requerimientos: se identifican necesidades mediante entrevistas a los usuarios.

2. Diseño conceptual: se da un esquema de alto nivel utilizando el modelo Entidad-Relación (ER).

3. Diseño lógico: se transforma el esquema conceptual en un modelo adecuado para un SGBD relacional.

4. Diseño físico: se especifican estructuras de almacenamiento y organización de archivos.

### Modelo Entidad-Relación (ER)

Este modelo, propuesto por Peter Chen en 1976, permite representar entidades, asociaciones y atributos de manera intuitiva. Se han desarrollado variaciones que mantienen sus conceptos básicos.

- **Entidades:** Representan objetos del mundo real con existencia propia. Se representan con rectángulos.
- **Asociaciones:** Interrelaciones entre entidades, representadas por rombos.
- **Cardinalidad:** Define el número de entidades que pueden estar asociadas (1:N, 1:N, N:N).
- **Atributos:** Características de las entidades/asociaciones, pueden ser simples, compuestos, mono/multivalores o derivados.
- **Identificadores:** Atributos que identifican de manera única a una entidad.

### Extensiones del modelo ER

1. roles: se usan cuando una entidad desempeña más de un papel en una asociación.
2. superclase y subclase (TISA): permite especializar o generalizar entidades.
3. Entidades fuertes y débiles: Las débiles dependen de otras para su existencia.

### Restricciones de consistencia

Las reglas de integridad garantizan la coherencia de los datos.

- Reglas implícitas: determinan la validez de asociaciones.
- Reglas adicionales: incluyen tipos de cardinalidad y restricciones específicas según el modelo de negocio.

## Construcción de software

### Notación del modelo Entidad-Relación y Restricciones Adicionales

Preguntas: ¿Cuando es más recomendable utilizar una entidad débil en lugar de una entidad fuerte dentro del diseño de una base de datos?

### Reglas de Traslado del modelo Entidad-Relación a Tablas

Para convertir un MER en un Modelo Relacional (MR), se siguen ciertas reglas que permiten traducir entidades, asociaciones, relaciones ISA, entidades débiles y roles en tablas.

Procedimiento general:

- Entidad  $\rightarrow$  Tabla: cada entidad se convierte en una tabla con el mismo nombre. Los atributos de la entidad son las columnas, y su identificador se vuelve la llave primaria. Si no hay identificador, se debe crear uno artificial.
- Asociación N:N  $\rightarrow$  Nueva tabla: se crea una nueva tabla que contiene las llaves primarias de las entidades participantes (llave simple compuesta) y los atributos propios de la asociación.
- Asociación 1:N  $\rightarrow$  Llave extranjera: NO se crea nueva tabla. Se agrega la llave primaria de la entidad del lado 1 a la tabla del lado N como llave foránea.
- Asociación 1:1  $\rightarrow$  Llave foránea en una tabla: Dista con añadir la llave primaria en la otra, en cualquier orden.

Casos Especiales:

- Relaciones ISA (Herencia): se tratan como relaciones 1:1. La entidad general comparte su llave primaria con las especies heredadas.
- Entidades débiles: Dependen de una entidad fuerte. Su tabla incluye como llave primaria tanto la llave de la entidad fuerte como un atributo que la distingue entre sí.
- Roles en relaciones reflexivas o múltiples: se crean las entidades generales, pero se renombran las columnas heredadas según el rol para evitar duplicación o ambigüedad.

Ejemplo de traducción:

Dado un MER con entidades A, B, C y asociaciones N:N y 1:N.

- A(91, 92, 93)
- B(61, b2, 91)
- C(61, c2, c3, 64)
- X(91, c1, x1, x2)  $\in$  Asociación N:N entre A y C)

Pregunta: ¿En qué casos conviene más usar una llave primaria artificial en lugar de una llave compuesta cuando se modela una relación N:N al modelo relacional?

## Construcción de Software

### Gestión de la comunicación en Proyectos (PMBOK)

La gestión de la comunicación es fundamental para asegurar que la información del proyecto fluya de manera eficiente entre todos los interesados. El módulo se basa en cinco procesos claves:

#### 1. Identificar los interesados:

Consiste en reconocer a las personas u organizaciones que serán afectadas por el proyecto, evaluando su nivel de influencia e impacto, y documentando sus intereses para diseñar estrategias de comunicación efectivas.

#### 2. Planificar la comunicación:

Se trata de definir quién necesita qué información, cuándo y cómo se debe entregar, y quién será el responsable. Esto busca asegurar la oportunidad, pertinencia y confidencialidad de la información.

#### 3. Distribuir información:

Involucra la entrega efectiva y continua de la información relevante conforme al plan. Se apoya en técnicas como presentaciones, reuniones, selección de medios y confidencialidad de la información, facilitación de procesos.

#### 4. Administrar las expectativas de los interesados:

Implica la comunicación constante para manejar inquietudes, influir positivamente en las expectativas, negociar y resolver conflictos durante todo el ciclo del proyecto.

#### 5. Reportar el desempeño:

Se refiere a entregar reportes que informen sobre el avance del proyecto, incluyendo análisis de desempeño, riesgos actuales, trabajo realizado, próximos pasos y cambios aprobados.

Pregunta: ¿Qué herramientas o estrategias recomienda para manejar a un interesado que inicialmente abora el proyecto pero cambia de postura a una resistencia activa durante su ejecución?

## Modelo Relacional y Álgebra Relacional

El modelo relacional, introducido por Edgar F. Codd a finales de los años 60, revolucionó el campo de las bases de datos al proporcionar una base teórica sólida basada en la teoría de conjuntos. Sus objetivos principales incluyen:

- Independencia física: La estructura lógica de los datos es independiente de su almacenamiento físico.
- Independencia lógica: Permite modificar la estructura de la base de datos sin afectar a los programas que aluden a ella.
- Flexibilidad: Facilita la presentación de datos de diversas formas según las necesidades del usuario.
- Uniformidad y sencillez: Ofrece estructuras de datos uniformes y lenguajes de consulta sencillos, facilitando su comprensión y uso.

## Construcción de Software

### Modelo Relacional Y Álgebra Relacional

En este modelo, el concepto central es la **relación**, representada como una tabla que consta de:

- **Dominios**: conjuntos de valores posibles para los atributos.
- **Tuplas**: filas de la tabla que representan instancias de datos.

Por ejemplo, considerando los dominios:

- Entrada: {COCKTAIL, entradas, ensaladas}
- Sopa: {crema, consomé, puré}
- Plato: {filete, pollo}

se puede definir una relación **Comida** (Entrada, sopa, plato), donde cada tupla es una combinación específica de estos elementos, como COCKTAIL, crema, filete).

El **Álgebra Relacional** proporciona un conjunto de operadores para manipular estas relaciones, permitiendo realizar consultas y transformaciones en la base de datos de manera formal y precisa.

Preguntar: ¿Por qué el Álgebra Relacional es fundamental para las bases de datos? Aplicar: Proporcionar un ejemplo práctico donde se utilicen operadores del álgebra relacional para resolver una consulta compleja en una base de datos real?

### Diagramas de secuencia con UML

Los diagramas de secuencia son una herramienta de modelado UML que representan cómo interactúan los objetos en un sistema a través del tiempo. Son esenciales en el diseño de software para visualizar el comportamiento dinámico de los sistemas, mostrando el orden de los mensajes entre objetos.

#### Conceptos clave:

- Diagramas de interacción: muestran objetos, relaciones y mensajes.
  - Si enfatizan el tiempo → Diagrama de secuencia
  - Si enfatizan estructura → Diagrama de colaboración
- Eje del diagrama:
  - Eje X: Objetos involucrados
  - Eje Y: Mensajes en orden temporal

#### Conceptos principales:

- Línea de vida: Línea vertical discontinua que indica la existencia de un objeto.
- Foco de control: Rectángulo que representa la ejecución de una acción por un objeto.
- Mensajes:
  - Invocaciones: flechas abiertas al comienzo hasta conectar la respuesta (flecha con cabecera abierta →)
  - Asignaciones: NO biyectivas; inicián un nuevo nido (flecha con cabecera abierta →)
  - Retornos: flechas discontinuas (queden omitidos si son evidentes) →)

## Construcción de software

### Diagramas de secuencia UML

TIPOS de diagramas de secuencia:

- Instancia: Escenario específico de ejecución.
- General: Describe posibles ramificaciones, bucles y condiciones de un caso de uso.

Fragments combinados (operadores):

- alt: Alternativa (if... else)
- opt: Opción condicional (cond o if simple)
- loop: Repetición (while)
- ref: Referencia a otro diagrama
- par: Ejecución en paralelo
- crit: Región crítica, solo un proceso se ejecuta a la vez

ESTOS fragmentos ayudan a implementar estructuras de control complejas dentro del diagrama.

### ENTRADAS SALIDAS

PREGUNTA: ¿En qué momento del proceso de diseño recomendaría utilizar diagramas de secuencia: antes o después de definir completamente los diagramas de clase, y por qué?

## Algebra Relacional y SQL Básico

### Algebra Relacional:

Es un lenguaje formal que utiliza operadores para manipular relaciones (tablas) en bases de datos. Los operadores básicos incluyen:

- Selección ( $\sigma$ ): Exige filas que cumplen una condición específica.
- Proyección ( $\pi$ ): Exige columnas específicas de una tabla
- Unión ( $\cup$ ): Combina filas de dos tablas compatibles
- Diferencia ( $\setminus$ ): Obtiene filas que están en una tabla pero no en otra.
- Producto cartesiano ( $\times$ ): Combina todos las filas de dos tablas.
- Join ( $\bowtie$ ): Combina filas de dos tablas basándose en una condición.

### SQL Básico:

SQL Structured Query Language es un lenguaje de consulta basado en álgebra relacional. Las instrucciones básicas incluyen:

- select: Equivale a la proyección, seleccionando columnas específicas.
- from: Especifica las tablas de las cuales se extraen los datos.
- where: Equivale a la selección, filtrando filas según condiciones.
- join: Combina filas de dos o más tablas basándose en una relación.

### Funciones Agrupadas en SQL

Funciones agrupadas comunes:

- COUNT(): Cuenta el número de filas.
- SUM(): Suma los valores de una columna numérica.
- AVG(): Calcula el promedio de una columna numérica.
- MAX(): Obtiene el valor máximo de una columna.
- MIN(): Obtiene el valor mínimo de una columna.

Estas funciones se utilizan comúnmente con la cláusula GROUP BY para agrupar resultados según una o más columnas y aplicar funciones agrupadas a cada grupo.

## CONJUNTO DE SOFTWARE

### FUNCIONES AGREGADAS EN SQL

Ej:  
SELECT departamento, AVG(salario)  
FROM empleado  
GROUP BY departamento;

Esta consulta devolverá el salario promedio por departamento. Además se pueden utilizar con la cláusula HAVING para filtrar grupos basándose en condiciones adicionales.

Preguntas: ¿Podría explicar como se implementa la operación de división del algoritmo SQL, considerando que no existe una instrucción similar para ello?

Métodología para diseñar casos de prueba a partir de casos de uso.

El artículo proponiendo para la lectura, indica cómo los casos de uso pueden ser utilizados estratégicamente para mejorar la planificación y ejecución de pruebas de software. Se parte de la premisa de que aunque los pruebas representan entre el 30% y el 50% de los costos de desarrollo, muchos productos siguen llegando al mercado con deficiencias, en gran medida porque el testing tradicionalmente comienza demasiado tarde y se realiza sin una metodología clara. La propuesta del autor es integrar el testing desde las primeras fases del desarrollo aprovechando los casos de uso - que definen los requisitos funcionales del sistema - para construir una base sólida de pruebas desde el principio.

Un caso de uso describe de manera textual y estructurada cómo un actor (persona o sistema externo) interactúa con el sistema para lograr un objetivo, incluyendo el flujo normal de eventos (lo que debería ocurrir) y los flujos alternativos (que tienen ante ellos o siguientes excepciones).

La metodología que se propone para generar casos de prueba consiste en tres pasos principales:

1. Generar escenarios de uso; cada escenario representa una posible ejecución de un caso de uso, considerando tanto el flujo básico como los alternativos. Esto asegura que no solo se prueben los caminos "felices" donde todo funciona correctamente, sino también condiciones anómalias o errores que deben manejar.
2. Identificar casos de prueba: para cada escenario identificado, se crea al menos un caso de prueba. Estos casos definen qué condiciones deben cumplirse para que el escenario se pueda ejecutar, especificando inputs, condiciones previas y resultados esperados. Es recomendable pensar en condiciones válidas e inválidas para asegurar una cobertura robusta.
3. Asignar datos concretos: Finalmente, se asignan valores reales a las condiciones especificadas en los casos de prueba, preparando así los datos necesarios para su ejecución efectiva. Esta etapa convierte descripciones generales en scripts listos para ser usados en pruebas automatizadas o manuales.

Nota: ¿Cómo podemos aplicar esta metodología basada en los casos de uso a proyectos existentes? Dado que los requisitos evolucionan constantemente y los casos de uso pueden cambiar a lo largo del desarrollo?

## Construcción de software

### Consultas en SQL usando roles y subconsultas

Uso de alias y sinónimos (roles):  
Cuando una tabla debe aparecer más de una vez en una consulta - como ocurre en una tabla de ciudades al representar tanto origen (como destino) en un viaje, se pueden usar alias para distinguir sus roles. Por ejemplo:

```
SELECT id_viaje, origen.nombre, destino.nombre, fecha  
FROM viajes, ciudades origen, ciudades destino  
WHERE viajes.id_origen = origen.id_ciudad  
AND viajes.id_destino = destino.id_ciudad;
```

Alternativamente se pueden usar sinónimos permanentes con CREATE SYNONYM para asignar nombres alternativos como origen y destino a la tabla ciudades, permitiendo una consulta más clara.

Este mismo patrón se usa en jerarquías, como cuando un empleado tiene un jefe (quien es parte de la misma tabla empleados), también se relativa con alias o sinónimos, permitiendo estructurar relaciones (recursivas).

### Comparaciones temporales:

Cuando se desea comparar registros de una misma tabla pero en diferentes fechas, como comparar ventas de productos en dos días distintos, se utilizan múltiples filas sobre la misma tabla. Esto permite hacer operaciones como diferencias de cantidades:

```
SELECT p.id_producto, p.descripcion, v1.cantidad - v2.cantidad AS diferencia  
FROM Productos p, Ventas diarias v1, Ventas diarias v2  
WHERE p.id_producto = v1.id_producto  
AND p.id_producto = v2.id_producto  
AND v1.fecha = '1-SEP-00'  
AND v2.fecha = '2-SEP-00';
```

### Subconsultas:

Hecho mediante parámetros:

- Encontrar elementos no relacionados (productos no vendidos) usando NOT IN o NOT EXISTS.
- Establecer condiciones adicionales para filtrar resultados (como productos con ventas mayores a un valor determinado).

### Ejemplo:

```
SELECT id_producto, descripcion  
FROM Productos p  
WHERE 10000 < (SELECT SUM(v.cantidad * p.precio)  
                  FROM Ventas diarias v  
                 WHERE v.id_producto = p.id_producto)
```

## CONSULTAS DE SUBTIPLÍCOS

CONSULTAS EN TABLAS USANDO ROLLOS Y SUBCONSULTAS

Los subconsultas pueden usarse con operadores como IN, EXISTS, > / < y los binarios, interpretándose como condiciones dentro del WHERE de una consulta principal.

Presentar: a) Que ventajas o desventajas hay entre usar roles dinámicos y roles permanentes para representar roles múltiples en una misma entidad, especialmente en sistemas grandes o complejos?

## Normalización

La normalización es un proceso fundamental en el diseño de bases de datos que busca estructurar la información de manera lógica y eficiente, evitando problemas como la redundancia y la inconsistencia de datos. Un modelo bien diseñado es más estable y fácil de mantener y se adapta mejor a futuros cambios.

### Objetivos de la normalización

- Representar adecuadamente las relaciones entre datos
- Facilitar las consultas y reportes.
- Simplificar la actualización, inserción y eliminación de datos.
- Evitar reestructuraciones frecuentes por nuevos requerimientos.

### Pasos para normalizar

1. Descomponer los datos en registros bidimensionales.
2. Eliminar dependencias parciales respecto a llave primaria.
3. Eliminar dependencias transitivas.

### Formas Normales

#### 1. Primera Forma Normal (1FN)

- Todos los valores en las celdas son atómicos (no repetidos ni agrupados).
- Columnas homogéneas, con nombres únicos y sin filas duplicadas.

#### 2. Segunda Forma Normal (2FN)

- Cumple con 1FN y todos los atributos <sup>no</sup> clave definidos completamente de la clave primaria.

#### 3. Tercera Forma Normal (3FN)

- Cumple con 2FN y no hay dependencias transitivas entre atributos no clave y la clave primaria.

#### 4. Cuarta Forma Normal (4FN)

- Cada determinante (atributo que determina a otros) debe ser una clave candidata.

#### 5. Quinta Forma Normal (5FN)

- Eliminar de dependencias de valores múltiples, que ocurren cuando un atributo tiene múltiples valores independientes respecto a otro.

#### 6. Sexta Forma Normal (6FN)

- Se enfoca en eliminar dependencias de tipo producto; busca asegurar que una tabla no puede dividirse en subtablas que luego no puedan reconstruirse adecuadamente. Preguntas: si podría explicar con más detalle como identificar una dependencia de tipo producto en la práctica y en qué casos reales se suele aplicar la sexta forma normal?