Unidad Didáctica 3: Sistemas de Gestión de Bases de Datos Relacionales

U.D. 3

UD 3. Sistemas de bases de datos

1.1. Esquemas y Niveles de Abstracción

Arquitectura ANSI/SPARC

Propuesta de arquitectura del grupo de estudio ANSI/SPARC (1977) para los SGBD: plantea la definición de la base de datos a tres niveles de abstracción:

- **Nivel interno** → Esquema interno

  Descripción de la BD en términos de su representación física

- **Nivel conceptual** → Esquema conceptual

  Descripción de la BD con independencia del SGBD

- **Nivel externo** → Esquema externo

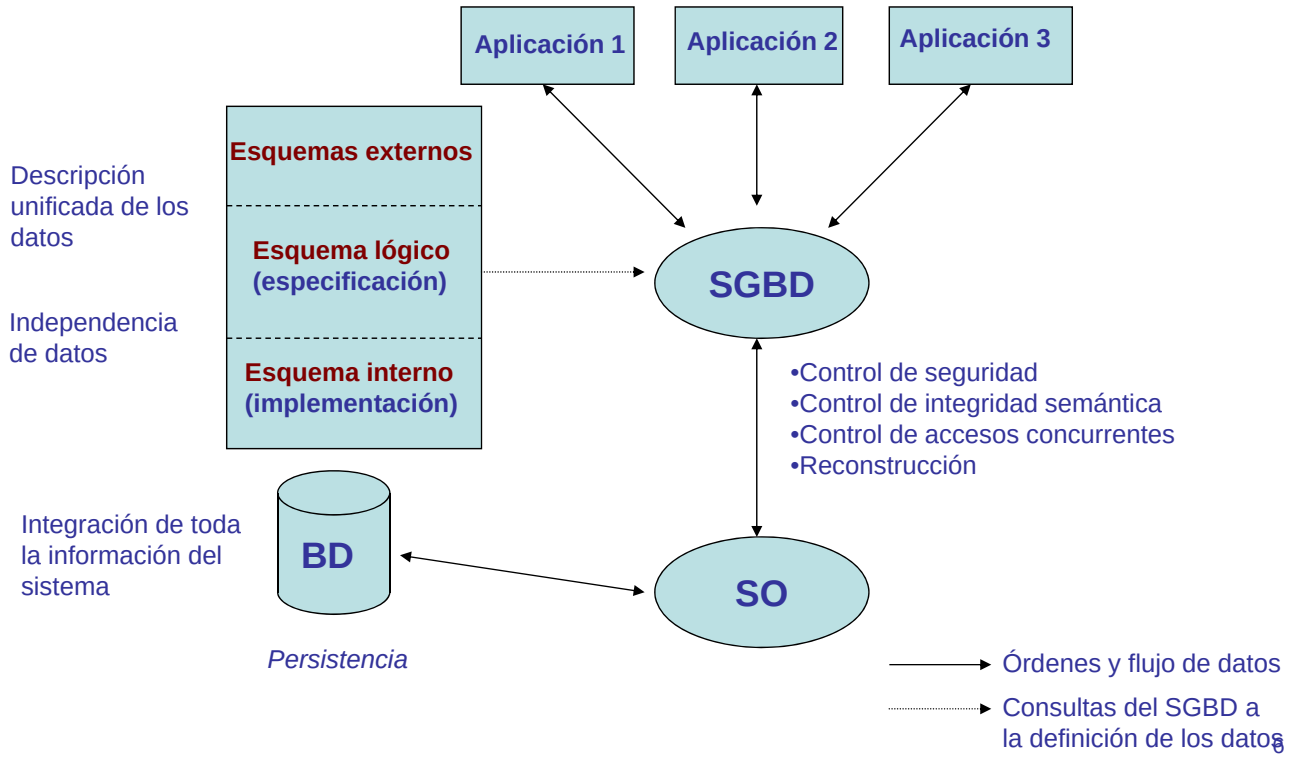
  Descripción de las vistas parciales de la BD que poseen los distintos usuarios

Arquitectura ANSI/SPARC

No existe un modelo conceptual generalizado y accesible a los distintos tipos de SGBD, se prefiere distinguir cuatro niveles:

- **Nivel conceptual** → Esquema conceptual (UD. 4)
  Descripción organizativa de la BD
- **Nivel lógico** → Esquema lógico (UD. 2)
  Descripción de la BD en términos del modelo de datos del SGBD
- **Nivel interno** → Esquema interno (no se ve en la asignatura)
  Descripción de la BD en términos de su representación física
- **Nivel externo** → Esquema externo (permisos y vistas)
  Descripción de las vistas parciales de la BD que poseen los distintos usuarios

Arquitectura ANSI/SPARC



Esquemas y Niveles de Abstracción

Esquema **externo** aplicación 1:

CREATE VIEW Administrativo (dni, nombre, salario_men)
AS SELECT dni, nombre, salario/14
FROM Empleado
WHERE tipo='AD'

Esquema **lógico**:

Empleado(dni, nombre, dirección, salario, tipo)
CP: {dni}

Esquema i**nterno**:

Fichero **ordenado** *Empleado* con **índice** primario sobre el campo *dni* en el **camino** *h:/disco1/gerencia*

Esquemas y Niveles de Abstracción

Aplicación 1: accede a la información a través del esquema externo 1

SELECT nombre, salario_men
FROM Administrativo
WHERE dni = *parámetro*

SGBD: control del acceso y resolución de la operación pedida

SO: Manipulación de los controladores de los dispositivos de memoria secundaria

Caso de estudio

Una pequeña **inmobiliaria** desea mantener información sobre los edificios cuya venta gestiona. Se quiere saber:

- De cada edificio, el código, la ubicación, el distrito, el propietario, el precio solicitado por éste y el agente encargado de la venta si ya está asignado.
- De cada propietario, el código, nombre y teléfono.
- De cada agente el DNI, el nombre, la comisión por cada venta, los años de antigüedad y el teléfono.

Las restricciones que deben cumplirse son las siguientes:

- La **comisión** de un agente no puede exceder el 3% si su antigüedad es menor de 3 años.
- No se quiere tener información de propietarios si no se tiene al menos un edificio para la venta.

Caso de estudio

Grupos de trabajo:

- El personal de **administración** tiene acceso a toda la información comentada.
- El **jefe** de la inmobiliaria sólo desea tener información referente a los edificios con precio solicitado superior a 5 millones. De cada uno desea el código, la ubicación, y el distrito.
- El jefe es el único que puede modificar la información de los agentes.

Caso de estudio

ESQUEMA LÓGICO:

```
CREATE SCHEMA Inmobiliaria

CREATE TABLE Edificios
    (Código d_cod PRIMARY KEY,
    Ubicación d_ubi NOT NULL,
    Distrito d_dis NOT NULL,
    Precio d_pre NOT NULL,
    Dni_age d_dni FOREIGN KEY REFERENCES Agente
        ON UPDATE CASCADE, ON DELETE NO ACTION
    Dueño d_cod NOT NULL,
    FOREIGN KEY(Dueño) REFERENCES Propietario (cod)
        ON UPDATE CASCADE ON DELETE CASCADE)
```


Caso de estudio

```
CREATE TABLE Propietarios
    (Cod d_cod PRIMARY KEY,
    Nombre d_nom NOT NULL,
    Teléfono d_tel NOT NULL)

CREATE TABLE Agentes
    (Dni_age d_dni PRIMARY KEY,
    Comisión d_com, Años d_años NOT NULL,
     Tel d_tel NOT NULL,
    CHECK NOT (años < 3 AND comisión > 3))

CREATE ASSERTION no_propet_sin_edificios
    CHECK NOT EXISTS
    (SELECT * FROM Propietarios WHERE cod NOT IN
                (SELECT Dueño FROM Edificio))
```

Caso de estudio

```
GRANT ALL ON Edificios TO PUBLIC;
GRANT ALL ON Propietarios TO PUBLIC;
GRANT SELECT ON Agentes TO PUBLIC;
```

ESQUEMA EXTERNO DEL JEFE:

```
CREATE VIEW más_de_5 AS
        SELECT código, ubicación, distrito
        FROM Edificios
        WHERE E.precio >= 5000000;
GRANT ALL ON más_de_5 TO Jefe;
GRANT ALL ON Agentes TO Jefe;
```

+ El resto de tablas del esquema lógico (excepto edificios)

ESQUEMA EXTERNO DEL PERSONAL ADMINISTRACIÓN :

Todas las tablas del esquema lógico

Caso de estudio

ESQUEMA INTERNO:

Edificios :

- Fichero disperso por dni_age
- Índice B+ sobre (distrito + precio)

Propietarios

- Fichero disperso por cod
- Índice B+ sobre nombre

Agentes

- Fichero desordenado (se suponen pocos agentes).

Independencia de datos

Un SGBD que soporte la arquitectura de niveles debe:

- Permitir definir los distintos esquemas de la base de datos (a excepción del esquema conceptual)
- Establecer las correspondencias entre los esquemas.
- Aislar los esquemas: los cambios en un esquema no deben afectar a los esquemas de nivel superior y por tanto, tampoco a los programas de aplicación.

INDEPENDENCIA DE DATOS (1.3)

1.2. Funcionamiento Básico de un SGBD

Funcionamiento Básico de un SGBD

SGBD: Software que permite la creación y manipulación de bases de datos.

Funcionamiento Básico de un SGBD

Los SGBD permiten:

- Descripción unificada de los datos e independiente de las aplicaciones
- Independencia de las aplicaciones respecto a la representación física de los datos
- Definición de vistas parciales de los datos para distintos usuarios
- Gestión de la información
- Integridad y seguridad de los datos

Funcionamiento Básico de un SGBD

Objetivos de técnicas BD	Funciones **SGBD**	Componentes SGBD
• Descripción unificada e independiente de los datos • Independencia de las aplicaciones • Definición de vistas parciales	Definición de datos a varios niveles: • esquema lógico • esquema interno • esquemas externos	Lenguajes de definición de esquemas y traductores asociados

Funcionamiento Básico de un SGBD

Objetivos de técnicas BD	Funciones **SGBD**	Componentes **SGBD**
Gestión de la información	Manipulación de los datos: • consulta • actualización Gestión y administración de la base de datos	Lenguajes de manipulación y traductores asociados Herramientas para: • reestructuración • simulación • estadísticas • impresión

Funcionamiento Básico de un SGBD

Objetivos de técnicas BD	Funciones SGBD	Componentes SGBD
Integridad y seguridad de los datos	Control de: • integridad semántica • accesos concurrentes • reconstrucción en caso de fallo • seguridad (privacidad)	Herramientas para: • control integridad • reconstrucción • control seguridad

Funcionamiento Básico de un SGBD

El jefe se pregunta:

¿código y ubicación de los edificios del distrito 05?

1. La aplicación interpreta la selección del jefe como:

SELECT código, ubicación
FROM más_de_5 WHERE distrito = '05';

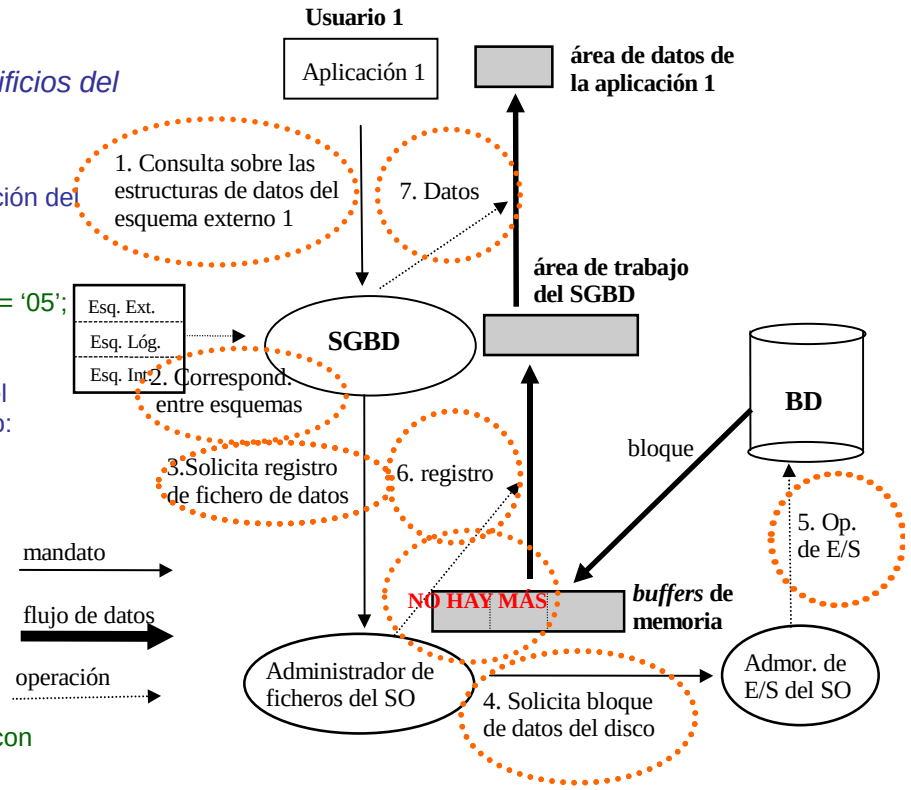
2. El SGBD convierte la consulta del esquema externo al esquema lógico:

SELECT código, ubicación
FROM Edificios E
WHERE E.precio >= 5000000
AND E.distrito = '05';

3,4,5,6. REPETIR:

"Leer usando el índice B+ sobre (distrito + precio) el primer registro con distrito = '05' y precio >= 5000000:

HASTA QUE NO HAYA MÁS REGISTROS



7. Eliminar los atributos que no se han solicitado

1.3. Independencia de datos.

Independencia de datos.

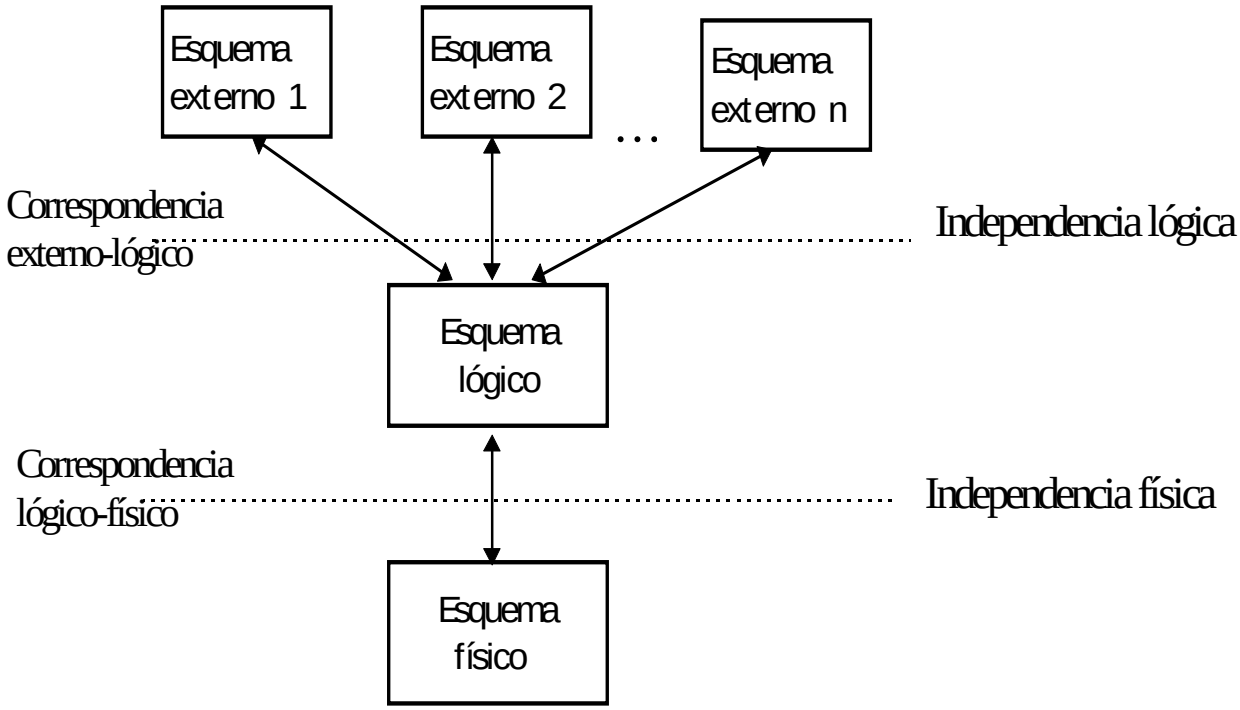
Propiedad que asegura que los programas de aplicación sean independientes de

los cambios realizados en datos que no usan

o

en detalles de representación física de los datos a los que acceden.

Independencia de datos.



Independencia lógica

Independencia lógica entre el esquema lógico y los externos:

Los esquemas externos y los programas de aplicación no deben verse afectados por modificaciones del esquema lógico sobre datos que no usan.

EJEMPLO:

Si al edificio se le añade un campo “Estado_de_conservación”, el esquema externo del jefe no cambia y la aplicación del jefe no se tiene que modificar.

Independencia física

Independencia física entre el esquema interno y el lógico:

El esquema lógico no debe verse afectado por cambios en el esquema interno referentes a la implementación de las estructuras de datos, modos de acceso, tamaños de páginas, caminos de acceso, etc.

EJEMPLO:

Si la relación edificio se cambia de localización física, el esquema lógico no se ve afectado.

Ligadura

Ligadura:

Transformación del esquema externo en el esquema interno.

Tipos
- Ligadura lógica (pasos 2 y 7).
- Ligadura física (pasos 3 y 6).

Cuando se produce la ligadura desaparece la independencia.

Hoyen día, en la mayoría de SGBD y entornos, la ligadura se realiza en cada acceso a la base de datos, con lo que los programas no han de alterarse mientras el esquema externo no se modifique.

UD 3. Sistemas de bases de datos

2. Transacciones, Integridad y Concurrencia

Objetivo de la tecnología de bases de datos

↓

Calidad de la información:

"*los datos deben estar estructurados reflejando adecuadamente los objetos, relaciones y las restricciones existentes en la parcela del mundo real que modela la base de datos*"

- Representación de los objetos, relaciones y restricciones en el esquema de la base de datos.
- Cambios en la realidad → Actualizaciones de los usuarios
- La información contenida en la base de datos debe preservar la definición del esquema.

2. Transacciones, Integridad y Concurrencia

Calidad de la información (perspectiva de la integridad):

- SGBD debe asegurar que los datos se almacenan correctamente
- SGBD debe asegurar que las actualizaciones de los usuarios sobre la base de datos se ejecutan correctamente y que se hacen permanentes

2. Transacciones, Integridad y Concurrencia

Herramientas del SGBD orientadas a la integridad:

- Comprobar (frente a actualizaciones) las restricciones de integridad del esquema
- Controlar la ejecución correcta de las actualizaciones (entorno concurrente)
- Recuperar (reconstruir) la base de datos en caso de pérdidas o accidentes

2.1. Concepto de Transacción

2.1. Concepto de Transacción

- La integridad de la base de datos se ve en peligro generalmente por las operaciones de acceso de las aplicaciones.

- Las operaciones de acceso a una base de datos se organizan en transacciones.

TRANSACCIÓN: Secuencia de operaciones de acceso a la base de datos que constituyen una unidad lógica de ejecución

Ejemplo de problema

Emp(dni, nombre, dir, dept)
  CP: {dni}
  CAj: {dept} → Dep(cod)

Dep(cod, nombre, ubicación)
  CP: {cod}

RI1: Todo departamento tiene al menos un empleado

Inserción de un nuevo departamento:
<“d2”, “Personal”, “Planta 3ª”>

cuyo primer empleado es el de *dni* 20

Ejemplo de problema

1ª Idea

1) Inserción en *Dep*: **<d2, “Personal”, “Planta 3ª>**

ERROR: la restricción *R1* no se cumple

2) Modificación de *Emp* en la tupla con *dni* 20

2ª Idea

1) Modificación de *Emp* en la tupla con *dni* 20

ERROR: la clave ajena sobre *dept* en *Emp* no se cumple

2) Inserción en *Dep*: **<d2, “Personal”, “Planta 3ª>**

Operaciones de definición de transacciones

principio:

Indica el comienzo de la transacción

fin:

Indica que se han terminado todas las operaciones de la transacción.

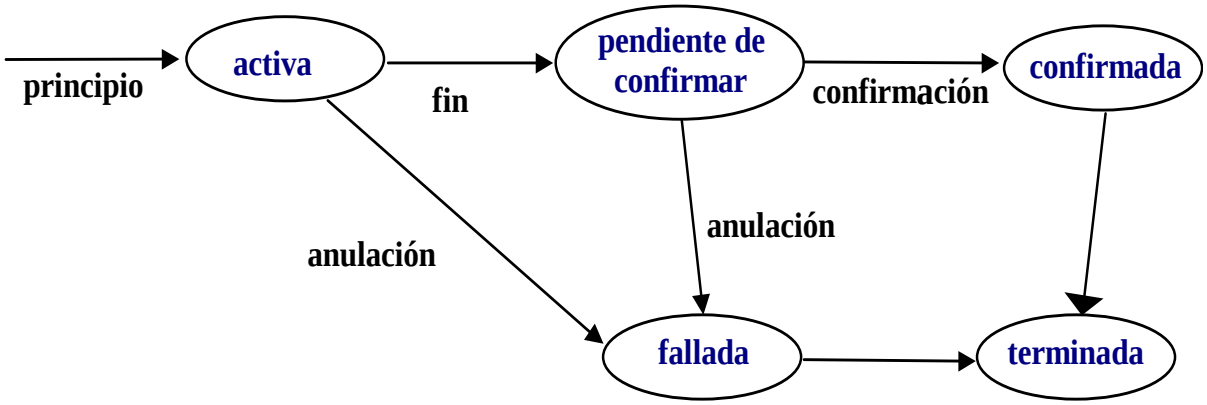
confirmación:

Indica el éxito de la transacción, permitiendo que el SGBD guarde los cambios efectuados en la base de datos

anulación:

Indica el fracaso de la transacción debido a algún motivo. El SGBD deshace todos los posibles cambios efectuados por la transacción

Estados de una transacción



Propiedades de una transacción: ACAP

- **Atomicidad:** una transacción es una unidad atómica de ejecución (o se ejecutan todas sus operaciones o ninguna)

- **Consistencia:** la transacción debe dar lugar a un estado de la base de datos consistente (se cumplen todas las restricciones de integridad)

- **Aislamiento:** las modificaciones introducidas por una transacción no confirmada no son visibles al resto de transacciones

- **Persistencia**: la confirmación implica la grabación de los cambios introducidos en la base de datos, de forma que no se puedan perder por fallo del sistema o de otras transacciones

Implementación de las transacciones

Depende del SGBD

- Actualización inmediata

  Las actualizaciones se realizan inmediatamente en la memoria secundaria. En caso de cancelación tienen que deshacerse

- Actualización diferida

  Las actualizaciones solo tienen efecto inmediato en memoria. Las actualizaciones se transfieren a la memoria secundaria cuando se confirman.

2.2. Integridad semántica

2.2. Integridad semántica

Restricción de integridad:

Propiedad del mundo real que modela la base de datos

- Las restricciones se definen en el esquema lógico y el SGBD debe velar por su cumplimiento.
- La comprobación se realiza cuando la base de datos cambia (se ejecuta una operación de actualización)
- Las restricciones que no se incluyen en el esquema de la base de datos se han de mantener en los programas de aplicación.

Esta situación es, por lo general, inadecuada si las restricciones son (o pueden ser) comunes a más de una aplicación, ya que se distribuye y diluye la responsabilidad de mantenerlas.

Tipo de restricción de integridad

- **Estáticas:** se deben cumplir en cada estado de la base de datos

  EJEMPLOS: Def. de Dominios, CP, CAj, VNN, UNIQUE, Assertions, ...

- **De transición:** se deben cumplir en dos estados consecutivos

  EJEMPLO: El precio de un inmueble no puede disminuir

Restricciones en SQL/92

- **Estáticas:**
  - sobre **dominios**: de valor
  - sobre **atributos**: valor no nulo, de rango, etc.
  - sobre **relaciones**: clave primaria, unicidad y claves ajenas.
  - sobre la **base de datos**: condiciones de búsqueda generales* (CREATE ASSERTION)
  - cuando se comprueba: después de cada operación (IMMEDIATE) / al final de la transacción (DEFERRED)
  - acciones compensatorias:
- **De transición:** se deben cumplir en dos estados consecutivos*

* (no suelen mantenerlas los sistemas comerciales)

Procedimientos de comprobación de la integridad: reglas de actividad, triggers, …

- Programación de la comprobación por parte del diseñador
- Permiten incluir en el esquema de la base de datos las restricciones complejas

- En los procedimientos se debe incluir:
  - Eventos u operaciones que los activan (evento y condición)
  - Código a ejecutar que incluye operaciones sobre la base de datos
  - Acciones de rechazo o compensación en caso de violación

Generalmente, las restricciones generales se implementan con:

- CREATE TRIGGER:
  Al indicar qué operaciones hay que controlar, se pueden definir muchos triggers en el sistema con una sobrecarga sobre el mismo menor

Y no con

- CREATE ASSERTION:
  Si el SGBD lo permite, puede tener un uso puntual, porque si se generaliza el sistema se enlentece muy significativamente al tener que ejecutar cada assertion para cualquier actualización de la base de datos.

Disparadores (triggers)

Los **triggers** (disparadores) introducen el concepto de reactividad, y tienen muchas otras aplicaciones aparte de la integridad:

- Mantenimiento de información derivada.
- Implementación de reglas de la organización.
- Administración de bases de datos (backups, avisos, etc.) y aspectos relacionados con seguridad (trazabilidad, registros, …)

2.3. Control de accesos concurrentes

2.3. Control de accesos concurrentes

Para mantener la integridad de la base de datos el SGBD debe controlar los accesos concurrentes:

- Evitando que los resultados de la ejecución de varios procesos (usuarios o programas) simultáneamente puedan llevar a resultados incorrectos, incoherentes o que se pierdan.

Ejemplo de problema

Cuentas

Nro.	Saldo
123	1000
555	2000

Cuentas

Nro.	Saldo
123	800
555	2000

Tiempo	P1	P2
t1	leer(123, saldo)	
t2		leer(123, saldo)
t3	saldo←saldo-100	
t4		saldo←saldo-200
t5	escribir(123, saldo)	
t6		escribir(123, saldo)

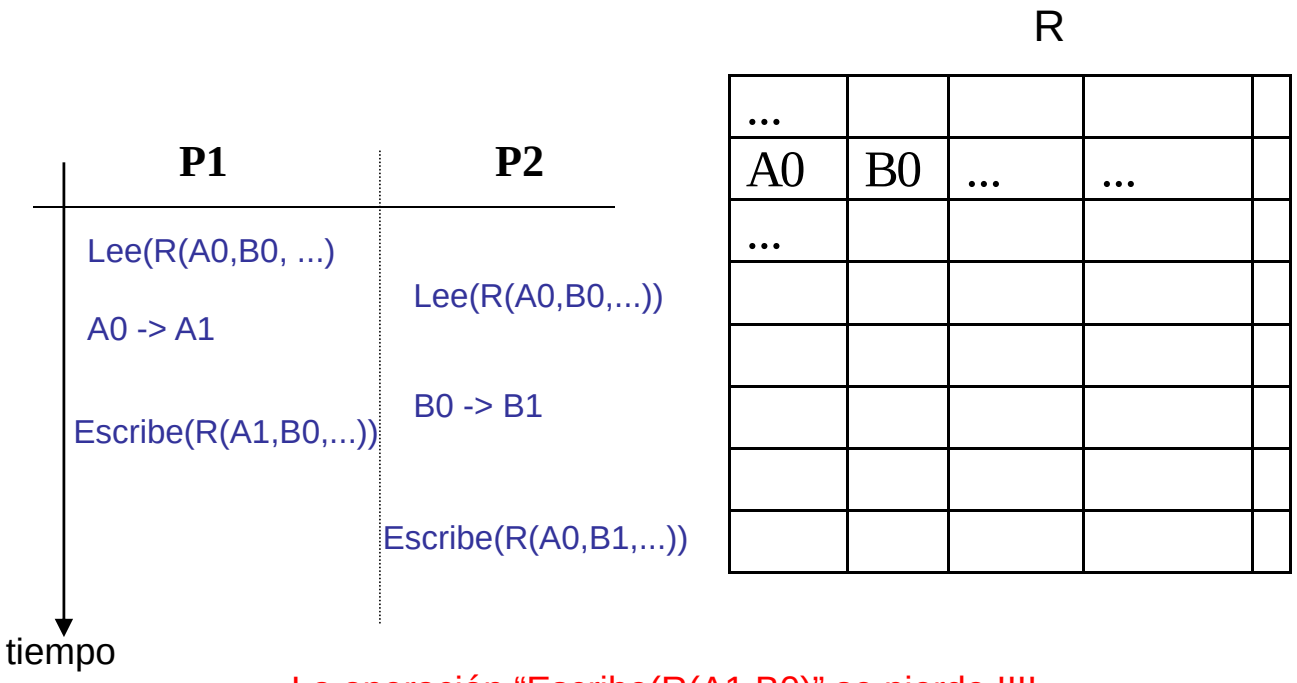
Posibles problemas

El SGBD debe controlar los accesos concurrentes de las aplicaciones.

Problemas por interferencia de accesos concurrentes:

a. Pérdida de actualizaciones,
b. Obtención de información incoherente correspondiente a varios estados válidos de la base de datos
c. Lectura de datos actualizados (no confirmados) que han sido sometidos a cambios que todavía pueden ser anulados.

A. Pérdida de actualizaciones



La operación “Escribe(R(A1,B0)” se pierde !!!!

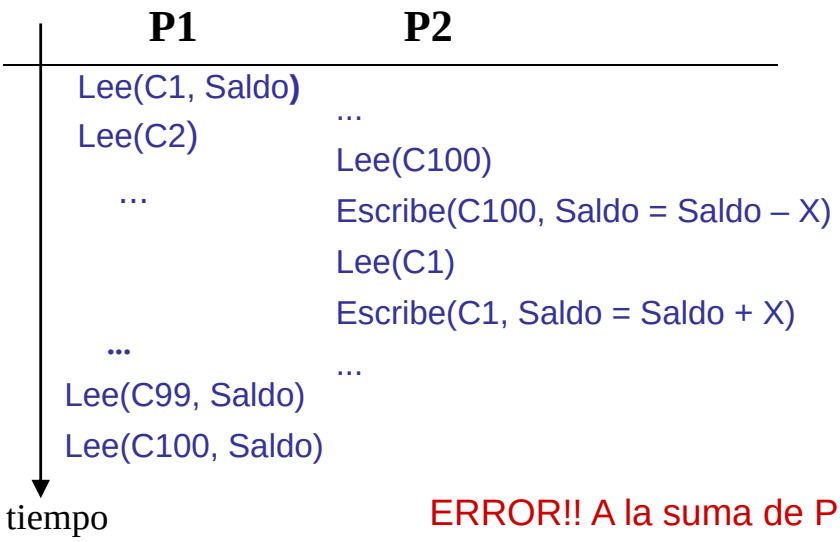
B. Obtención de información incoherente

P1: Obtención del total de saldos.

P2: Transferencia de la cuenta 100 a la 1.

Cuentas Corrientes

C1	€200000	
C2	...	..
C100	€200000	..

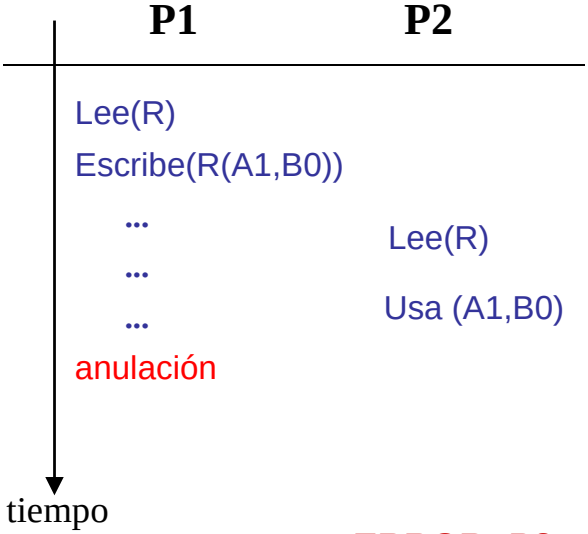


ERROR!! A la suma de P1 faltan X Euros.

C. Lectura de datos actualizados sin confirmar

R

...				
A0	B0	...	...	
...				



ERROR: P2 usa un dato inválido.

Técnicas

Reserva de Ocurrencias de Datos (Locks)

- Ejemplos a) y c) se reserva un registro.
- Ejemplo b) se reservan todos.
- Necesidad de controlar bloqueos (*deadlocks*)

Otras soluciones (para el ejemplo c): anulación en cascada o aislamiento de transacciones.

UD 3. Sistemas de bases de datos

Recuperación y seguridad

Hay dos aspectos irrenunciables en la tecnología actual de bases de datos:

- Recuperación:
  (Parte de la integridad, pero no desde el punto de vista de la consistencia):
  Una base de datos ha de poderse recuperar ante prácticamente cualquier tipo de fallo.

- Seguridad:
  Una base de datos no puede tener accesos no autorizados.

3.1. Reconstrucción de la BD

Implementación de las transacciones

La implementación depende del SGBD:

- Actualización inmediata
  Las actualizaciones se realizan inmediatamente en la memoria secundaria. En caso de cancelación tienen que deshacerse

- Actualización diferida
  Las actualizaciones solo tienen efecto inmediato en memoria. Las actualizaciones se transfieren a la memoria secundaria cuando se confirman.

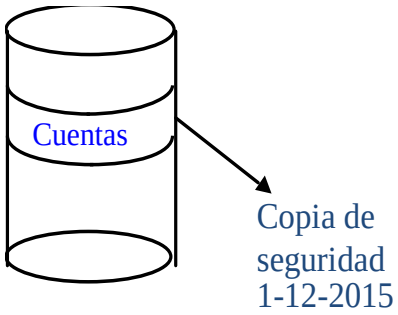
Reconstrucción de la BD

Las propiedades de atomicidad y persistencia de una transacción obligan al SGBD a asegurar que:

- Si se confirma, los cambios efectuados se graban en la base de datos y no se pierdan.

- Si se anula, los cambios efectuados sobre la base de datos se deshacen.

Reconstrucción de la BD



1-12-2015: Actualización de cuentas

transacción nro. 51: ¡fallo del sistema!

Procedimiento de recuperación:

- sustituir el fichero de Cuentas por su copia de seguridad

Efecto negativo:

- se han perdido las actualizaciones de 50 transacciones

Reconstrucción de la BD

- Las copias de seguridad, por sí mismas, no son la solución al problema de recuperación.
  - El aumento de la frecuencia de copias de seguridad no es una solución viable.
- La tecnología de bases de datos proporciona técnicas mucho más eficientes y robustas de cara a la reconstrucción de la base de datos.

La pérdida de datos "confirmados" es inadmisible con la tecnología actual.

Causas del fallo de una transacción

1. Locales a la transacción (funcionamiento del sistema normal)
   - Errores en la transacción (acceso a la base de datos incorrecto, cálculos fallidos, etc.)
   - Excepciones (violación de la integridad, de la seguridad, etc.)
   - Control de la concurrencia (estado de bloqueo entre dos transacciones)
   - Decisiones humanas (por programa o explícitas).

2. Externas a la transacción (errores del sistema)
   A. Fallos del sistema con pérdida de la memoria principal.
   B. Fallos del sistema de almacenamiento con pérdida de la memoria secundaria.

A. Pérdidas de memoria principal

- En el espacio de tiempo entre la confirmación de una transacción y la grabación de sus campos en memoria secundaria.
- La transacción está confirmada y sus cambios están en los bloques de los buffers.
- En dicho intervalo se produce un fallo con pérdida de memoria principal y los bloques de los buffers se pierden.

B. Pérdidas de memoria secundaria

- Transacción confirmada cuyos cambios están grabados en la base de datos.
- Fallo en la memoria secundaria y estos cambios se pierden.

Reconstrucción de la BD

Reconstrucción frente a fallos del sistema

Funciones
- Recuperar transacciones confirmadas que no han sido grabadas.
- Anular transacciones que han fallado.

- Módulo de reconstrucción.
  - Técnica más extendida: uso del *fichero diario (log o journal).*

Reconstrucción de la BD

Actividades sobre el fichero diario

- Registrar las operaciones de actualización de las transacciones.
- Se almacena en disco para evitar la desaparición por un fallo del sistema.
- Se graba periódicamente a una unidad de almacenamiento masiva.

Tipos de entradas

Tipo de entradas que se graban en el fichero diario

- [inicio, T]: se ha iniciado la transacción de identificador *T*.
- [escribir, T, X, valor_antes, valor_después]: la transacción *T* ha realizado una operación de actualización sobre el dato *X*.
- [leer, T, X]: la transacción *T* ha leído el dato *X*.
- [confirmar, T]: la transacción *T* ha sido confirmada.
- [anular, T]: la transacción *T* ha sido anulada.

Reconstrucción de la BD

*Supondremos **ACTUALIZACIÓN INMEDIATA***

Fallo de una transacción *T* → Deshacer cambios de *T*

- actualizar los datos modificados por T con su valor original (*valor_antes*).
- Buscar las entradas en el diario

  [escribir, T, X, valor_antes, valor_después]

Fallo del sistema → Aplicar el proceso anterior a todas las transacciones sin confirmar

Reconstrucción de la BD

Fallo del sistema

- Transacciones sin confirmar

  [inicio, T] en el diario sin *[confirmar, T]*
- Proceso anterior

- Transacciones confirmadas

  [confirmar, T]
- Volver a ejecutarlas:

  [escribir, T, X, valor_antes, valor_después]

Reconstrucción de la BD

PROBLEMAS:

- Tamaño del fichero diario puede crecer muy rápidamente.
- Recuperación en caso de fallo muy costosa (hay que rehacer muchas operaciones).

SOLUCIÓN:

Puntos de verificación (checkpoints)

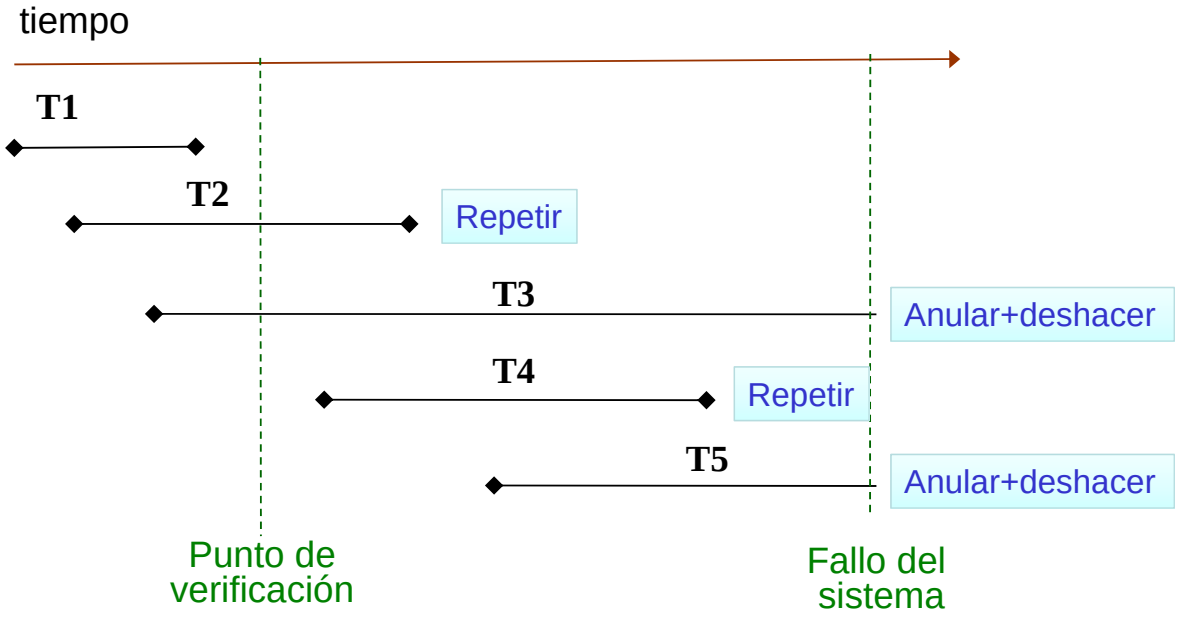
Reconstrucción de la BD

Puntos de verificación → Se graban en el diario periódicamente

- Suspender temporalmente la ejecución de transacciones.
- Grabar en el diario el punto de verificación.
- Forzar la grabación de todas las actualizaciones de las transacciones confirmadas (copiar los buffers a disco).
- Reanudar la ejecución de las transacciones suspendidas.

Reconstrucción de la BD

Puntos de verificación → Reconstrucción a partir del último



Reconstrucción de la BD

Reconstrucción frente a fallos del sistema de almacenamiento

- Pérdida de memoria secundaria.
- Base de datos puede estar dañada total o parcialmente.
- Técnica: reconstruir la base de datos a partir de
  - La copia de seguridad más reciente.
  - A partir del instante de la copia utilizar el diario para rehacer las operaciones realizadas por las transacciones confirmadas.

Reconstrucción de la BD

*Caso **ACTUALIZACIÓN DIFERIDA***

El mecanismo de reconstrucción es el mismo (las confirmadas se deben repetir), exceptuando:

- Las no confirmadas no deben ser deshechas.

3.2. Seguridad

3.2. Seguridad

Objetivo:

Sólo pueden acceder a la información las personas y procesos autorizados y en la forma autorizada.

Técnicas

- Identificación del usuario.
- Determinación de los accesos permitidos:

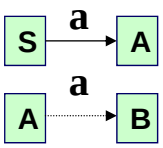
Modos

  - Lista de autorizaciones (objeto y operaciones permitidas) por usuario.
  - Niveles de autorización (menos flexible).

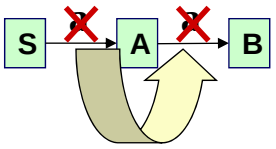
- Gestión de autorizaciones transferibles: traspaso de autorizaciones de un usuario a otro.

Requisitos para realizar la gestión de autorizaciones transferibles:

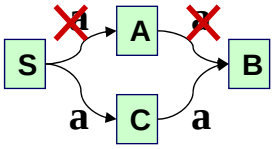
- Conocimiento de las autorizaciones de acceso de cada usuario (cuáles son transferibles a terceros y cuáles no).
- Transferencia de una autorización de un usuario a otro (en modo transferible o no).



- Revocación posterior de una autorización de acceso:
  - Si se otorgó en modo transferible, revocación de las autorizaciones que partieron de ella.



Revocación independiente de una autorización de acceso otorgada de forma múltiple.



B debe conservar el permiso 'a'

Gestión de autorizaciones en SQL

Gestión de autorizaciones en SQL:

definición_privilegio::= **GRANT**
{ ALL |
SELECT |
INSERT [(nom_atr$_1$,…, nom_atr$_n$)] |
DELETE |
UPDATE [(nom_atr$_1$,…, nom_atr$_n$)]
}
ON nom_relación **TO** {usuario$_1$,…, usuario$_m$ | PUBLIC}
[WITH GRANT OPTION]

- Con la cláusula PUBLIC se otorgan los privilegios a todos los usuarios

Gestión de autorizaciones en SQL

- Los privilegios otorgados son:
  - SELECT: permite que el usuario consulte la relación.
  - INSERT [(nom_atr1,…, nom_atrn)]: permite que el usuario inserte tuplas en la relación dando valor sólo a los atributos especificados.
  - DELETE: permite que el usuario elimine tuplas de la relación.
  - UPDATE [(nom_atr1,…, nom_atrn)]: permite que el usuario modifique el valor los atributos especificados.
  - ALL: se otorgan todos los privilegios anteriores.
- La cláusula WITH GRANT OPTION otorga permiso para ceder a terceros los privilegios obtenidos.
- Existe también una instrucción para quitar privilegios (instrucción REVOKE).

Seguridad

Privacidad, seguridad e implicaciones éticas y legales:

- Proteger el acceso y difusión de datos personales por usuarios no autorizados.
- Prohibir la cesión de datos a terceros de carácter personal o información aparentemente agregada (consultas parametrizadas) que puedan desvelar información particular.
- En algunos caso es necesario comunicar la misma creación de una base de datos (en España, la Agencia de Protección de Datos.
- Custodia de copias de seguridad, discos duros retirados, etc.
- Celo en el cuidado de la contraseña de administrador.