Unidad Didáctica 3:
Sistemas de Gestión de Bases de Datos

(Doc. UD3)

UD3.- Sistemas de Gestión de Bases de Datos

1.- Arquitectura ANSI/SPARC
- 1.1.- Esquemas y niveles de abstracción
- 1.2.- Funcionamiento básico de un SGBD
- 1.3.- Independencia de datos

2.- Transacciones, integridad y concurrencia
- 2.1.- Concepto de transacción
- 2.2.- Integridad semántica
- 2.3.- Control de accesos concurrentes

3.- Recuperación y Seguridad
- 3.1.- Reconstrucción de la base de datos
- 3.2.- Seguridad

1.- 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:

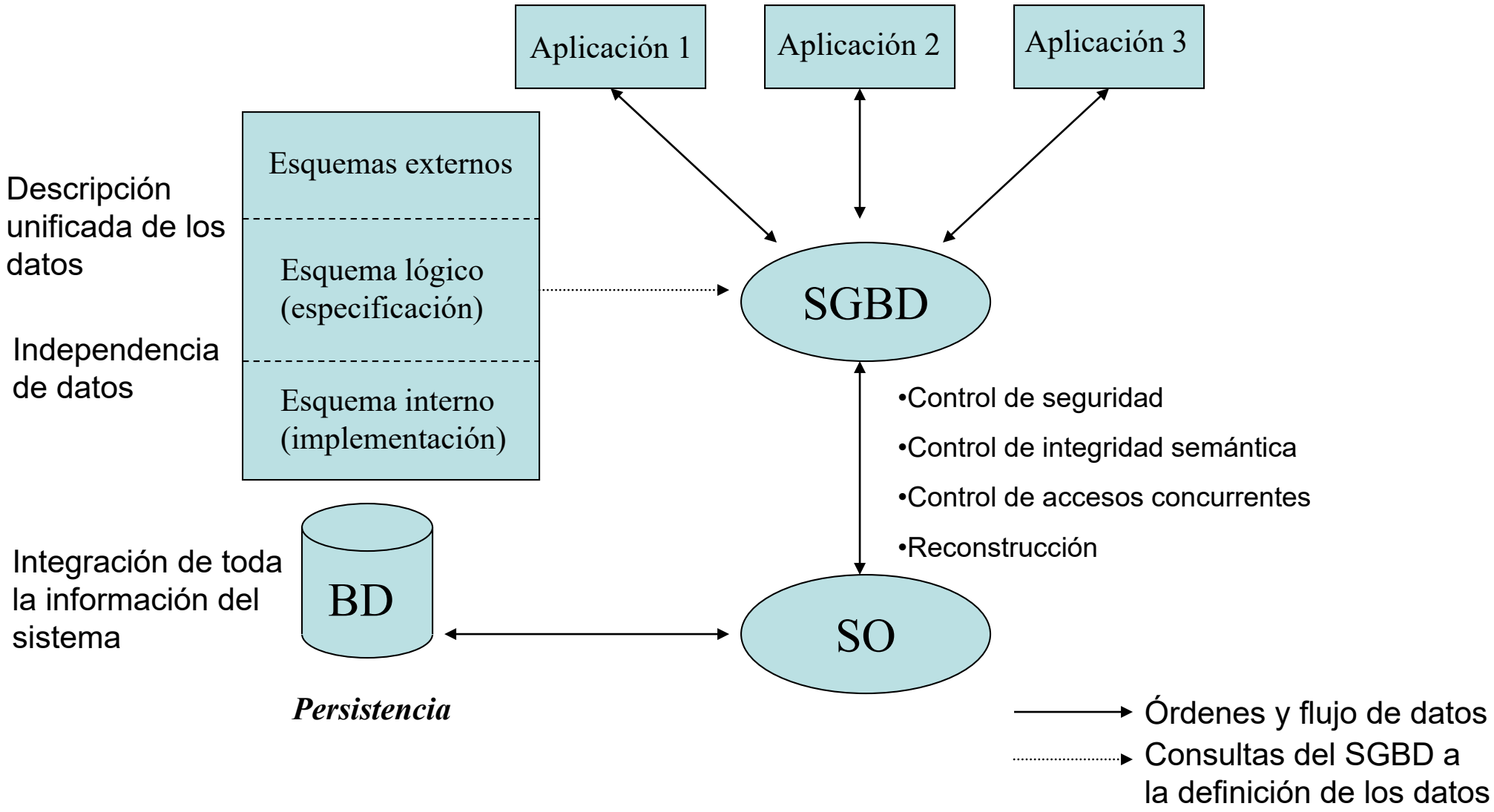
- **Nivel conceptual** → Esquema conceptual
  - descripción de la BD con independencia del SGBD
- **Nivel interno** → Esquema interno
  - descripción de la BD en términos de su representación física
- **Nivel externo** → Esquema externo
  - descripción de las vistas parciales de la BD que poseen los distintos usuarios

1.- Arquitectura ANSI/SPARC

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

- **Nivel conceptual** → Esquema conceptual (U.D. 4)
  - descripción organizativa de la BD
- **Nivel lógico** → Esquema lógico (U.D. 2 y U.D. 4)
  - descripción de la BD en términos del modelo de datos del SGBD
- **Nivel interno** → Esquema interno (no lo veremos 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

1.- Arquitectura ANSI/SPARC



1.1.- Esquemas y Niveles de Abstracción: Ejemplo

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.

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 Conceptual

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

Esquema Lógico

Propietario (cod: d_cod, nombre: d_nom, telefono: d_tel)

CP:{cod} VNN:{nombre, telefono}

Agente(dni: d_dni, comision: d_com, anyos: d_anyos, tel: d_tel)

CP:{dni} VNN:{comision, anyos, tel} RI: Si anyos < 3 → comision <=3

Edificio(cod: d_cod, ubicacion: d_ubi, distrito: d_dis, precio: d_pre, agente: d_dni, propietario: d_cod)

CP:{cod} VNN: {ubicacion, distrito, precio, propietario}

CAj: {agente} → Agente(dni)

Modificación en cascada, Borrado restrictivo

CAj: {propietario} → Propietario(cod)

Modificación en cascada, Borrado en cascada

$RI_{general}$: Todo propietario tiene un edificio

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

```
CREATE TABLE Propietario (
    cod d_cod CONSTRAINT pk_propietario PRIMARY KEY,
    nombre d_nom NOT NULL,
    telefono d_tel NOT NULL
);
CREATE TABLE Agente (
    dni d_dni CONSTRAINT pk_agente PRIMARY KEY,
    comision d_com,
    anyos d_años NOT NULL,
    tel d_tel NOT NULL,
    CHECK NOT (anyos < 3 AND comision > 3)
);
```

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

```
CREATE TABLE Edificio (
    cod d_cod CONSTRAINT pk_edificio PRIMARY KEY,
    ubicacion d_ubi NOT NULL,
    distrito d_dis NOT NULL,
    precio d_pre NOT NULL,
    agente d_dni CONSTRAINT fk_edificio_agente REFERENCES Agente
        ON UPDATE CASCADE ON DELETE NO ACTION
    propietario d_cod NOT NULL
        CONSTRAINT pk_edificio_propietario REFERENCES Propietario (cod)
            ON UPDATE CASCADE ON DELETE CASCADE
);
```

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

```
CREATE ASSERTION no_propietario_sin_edificios CHECK
      NOT EXISTS (
            SELECT *
            FROM Propietario
            WHERE codi NOT IN (SELECT propietario FROM Edificio)
);
GRANT ALL ON Edificio TO PUBLIC;
GRANT ALL ON Propietario TO PUBLIC;
GRANT SELECT ON Agente TO PUBLIC;
```

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

Esquema Externo Jefe

```
CREATE VIEW mas_de_5 AS
        SELECT codi, ubicacion, distrito
        FROM Edificio
        WHERE E.precio >= 5000000;
GRANT ALL ON mas_de_5 TO Jefe;
GRANT ALL ON Agentes TO Jefe;
```

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

Esquema Externo del personal de administración

Todas las tablas del esquema lógico

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

Esquema Interno

Edificio

Fichero disperso por dni

Índice B+ sobre (distrito + precio)

Propietario

Fichero disperso por codi

Índice B+ sobre nombre

Agente

Fichero desordenado (se suponen pocos agentes).

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

El jefe se pregunta:

¿código y ubicación de los edificios del distrito 05?

1. La aplicación interpreta la selección del jefe como:

SELECT codi, ubicacion

FROM más_de_5 WHERE distrito = '05';

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

SELECT codi, ubicacion

FROM Edificio E

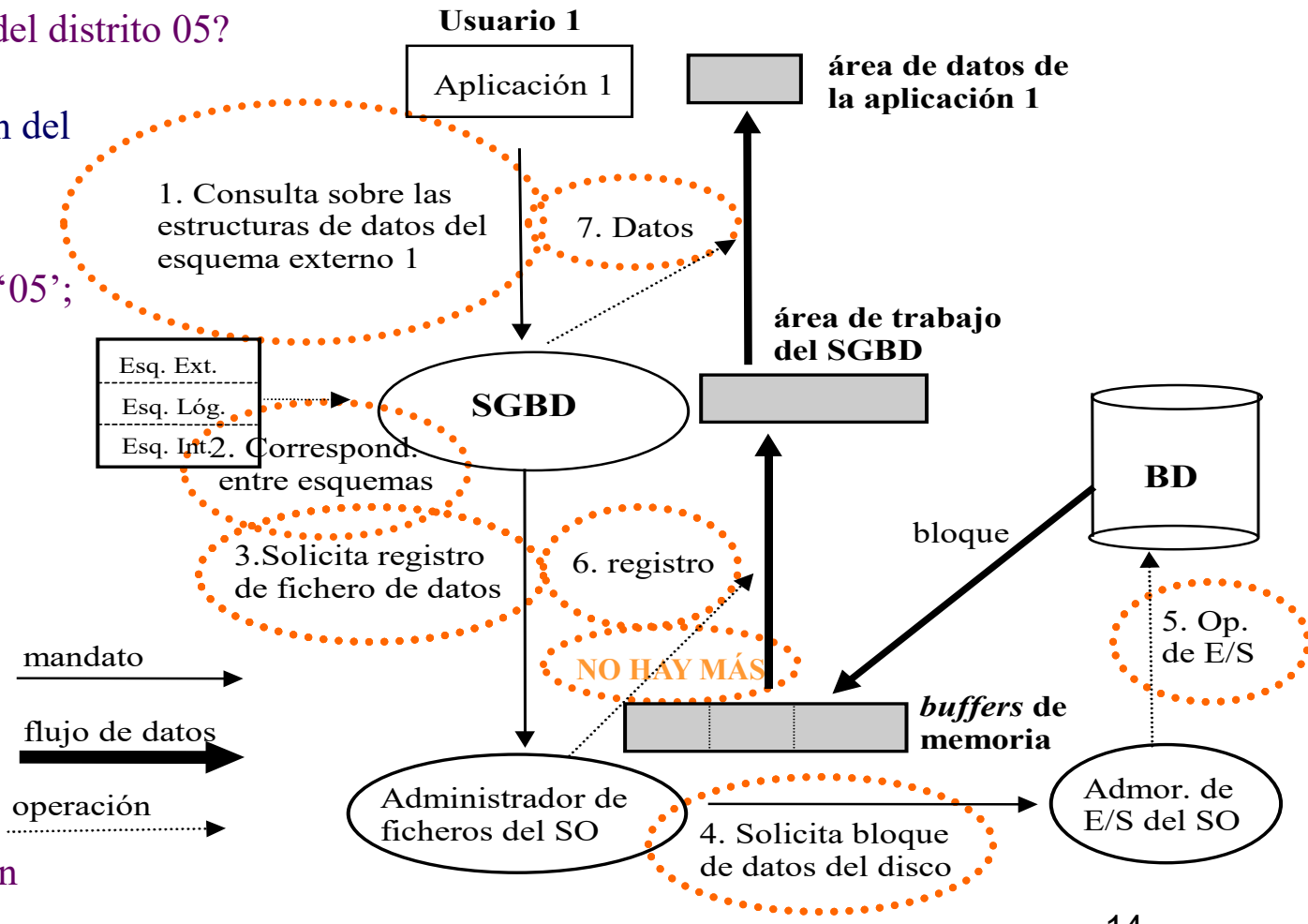
WHERE E.precio >= 5000000

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



14

1.1.- Esquemas y Niveles de Abstracción: Ejemplo

Un SGBD que soporte la arquitectura de niveles debe:

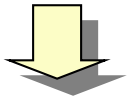
- permitir definir los distintos esquemas de la base de datos (a excepción del esquema conceptual)
- establecer las correspondencias entre los esquemas.
- aislar los esquemas: los cambios en un esquema no deben afectar a los esquemas de nivel superior y por tanto, tampoco a los programas de aplicación.

INDEPENDENCIA DE DATOS (3.1.3)

1.2.- Funcionamiento Básico de un SGBD

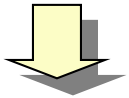
SGBD: Software que permite la creación y manipulación de bases de datos.

SGBD



Se basa

modelo de datos (modelo relacional)



Se compone

estructuras de datos y operadores asociados

1.2.- Funcionamiento Básico de un SGBD

Objetivos de las téc. de BD	Funciones del SGBD	Componentes del SGBD
• Descripción unificada e independiente de los datos • Independencia de las aplicaciones • Definición de vistas parciales	Definición de la BD a varios niveles: • esquema lógico • esquema interno • esquemas externos	Lenguajes de definición de esquemas y traductores asociados
• Gestión de la información	Manipulación de los datos: consulta y actualización. Gestión y administración de la base de datos.	Lenguajes de manipulación y traductores asociados. Herramientas para: • reestructuración • simulación • estadísticas • impresión
• 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

1.3.- 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**.

1.3.- Independencia de datos

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.

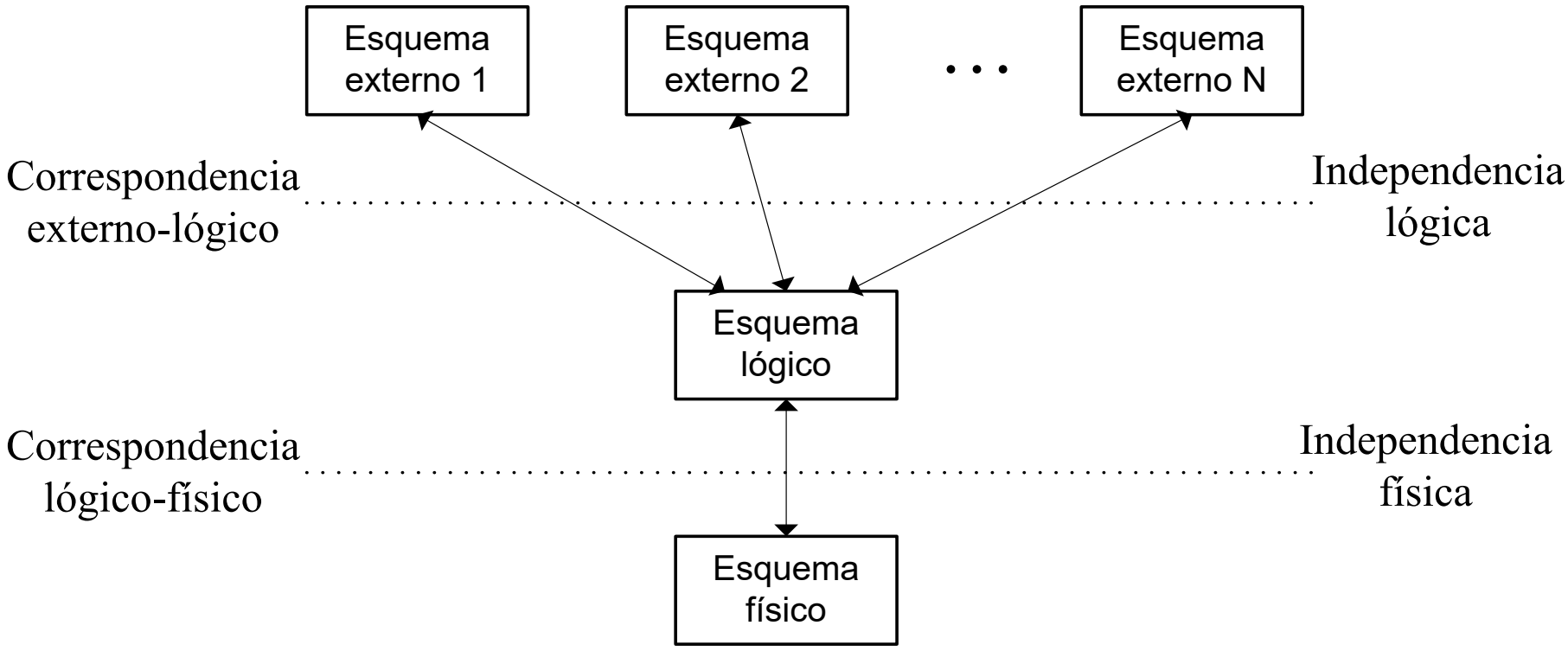
  EJEMPLO: Si al edificio se le añade un atributo “estado_de_conservación”, el esquema externo del jefe no cambia y la aplicación del jefe no se tiene que modificar.

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.

1.3.- Independencia de datos



2.- Transacciones, Integridad y Concurrencia

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"

- Es un objetivo de la tecnología de bases de datos
- Representación de los objetos, relaciones y restricciones en el esquema de la base de datos.
- Cambios en la realidad $\rightarrow$ 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)

- El SGBD debe asegurar que los datos se almacenan correctamente
- El 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.- Transacciones, Integridad y Concurrencia

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

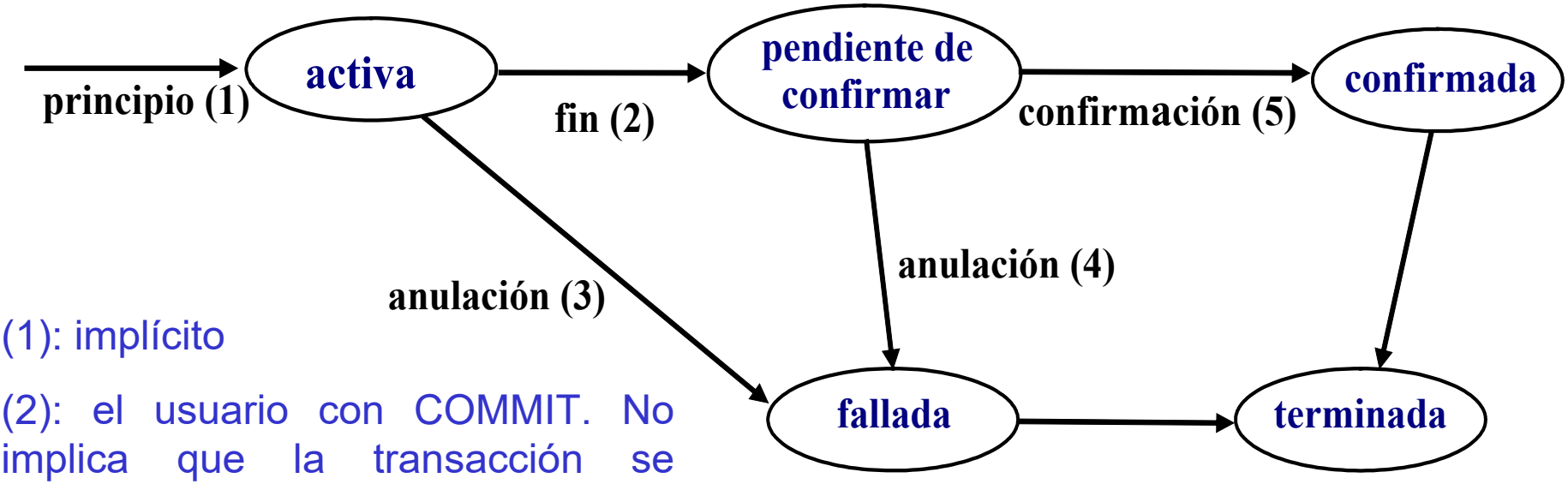
Las operaciones de acceso a una base de datos se organizan en transacciones.

TRANSACCIÓN

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

2.1.- Concepto de Transacción

Operaciones de definición de las transacciones:



- (1): implícito
- (2): el usuario con COMMIT. No implica que la transacción se confirme definitivamente.
- (3): usuario con ROLLBACK o el SGBD por un error mientras se ejecuta la transacción.
- (4) y (5): el SGBD después de realizar comprobaciones de integridad, concurrencia, etc.

- Fallada: ningún cambio hecho por la transacción puede permanecer en la BD.
- Confirmada: todos los cambios hecho por la transacción deben permanecer en la BD.

2.1.- Concepto de Transacción

Propiedades que deben cumplir las transacciones (ACID):

- **A**tomicity (atomicidad): una transacción es una unidad atómica de ejecución (o se ejecutan todas sus operaciones o ninguna)
- **C**onsistency (consistencia): la transacción debe dar lugar a un estado de la base de datos consistente (se cumplen todas las restricciones de integridad)
- **I**solation (aislamiento): las modificaciones introducidas por una transacción no confirmada no son visibles al resto de transacciones
- **D**urability (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

2.1.- Concepto de Transacción

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

Una restricción de integridad es una propiedad del mundo real del cual la base de datos es una representación. Para que esta representación sea consistente con la realidad, la base de datos debe satisfacer las restricciones en cualquier instante.

- Las restricciones de integridad incluidas en el esquema lógico de la base de datos son responsabilidad del SGBD cuando la base de datos cambia.
- Las restricciones de integridad no incluidas en el esquema de la base de datos son responsabilidad de los programas de acceso a la base de datos.

2.2.- Integridad semántica

Tipos de restricciones de integridad:

- **estáticas**: se deben cumplir en cada estado de la base de datos (representable en expresiones lógicas)
  - Restricciones sobre atributos: de valor, incluido VNN
  - Restricciones sobre relaciones: clave primaria, unicidad y claves ajenas.
  - Restricciones generales: no soportadas por los SGBD

EJEMPLO: la edad de un alumno es un número natural mayor que 16

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

EJEMPLO: la edad de un alumno no puede decrecer

2.3.- Control de accesos concurrentes

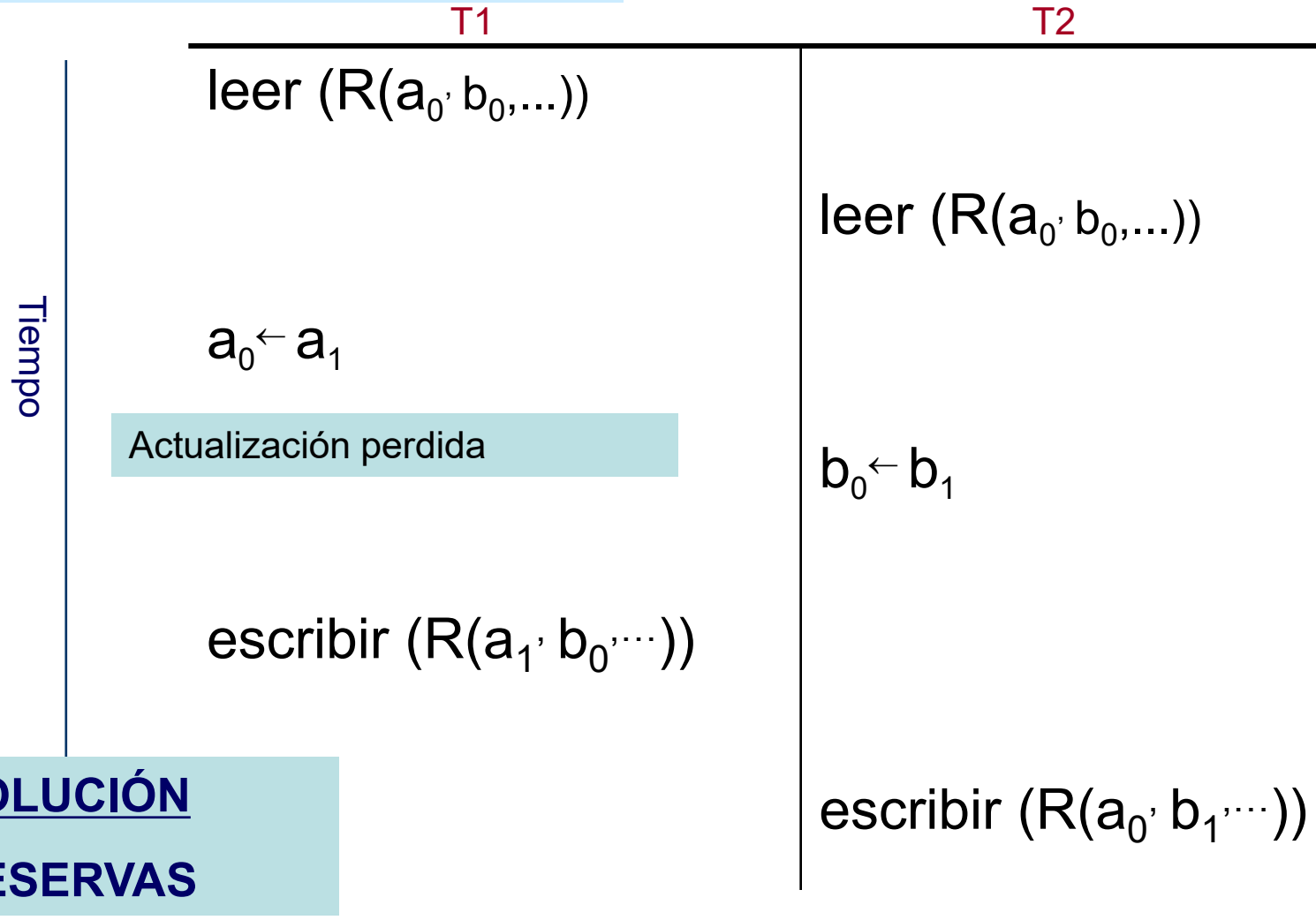
Para mantener la integridad de la base de datos en un entorno concurrente, el SGBD debe evitar que los resultados de la ejecución de una transacción sean incorrectos, incoherentes o se pierdan debido a la ejecución simultánea de otra transacción.

Algunos problemas:

- Pérdida de actualizaciones.
- Obtención de información incoherente de estados válidos.
- Lectura de dates actualizados (no confirmados) que han sido sometidos a cambios que se pueden anular.

2.3.- Control de accesos concurrentes

Pérdida de actualizaciones

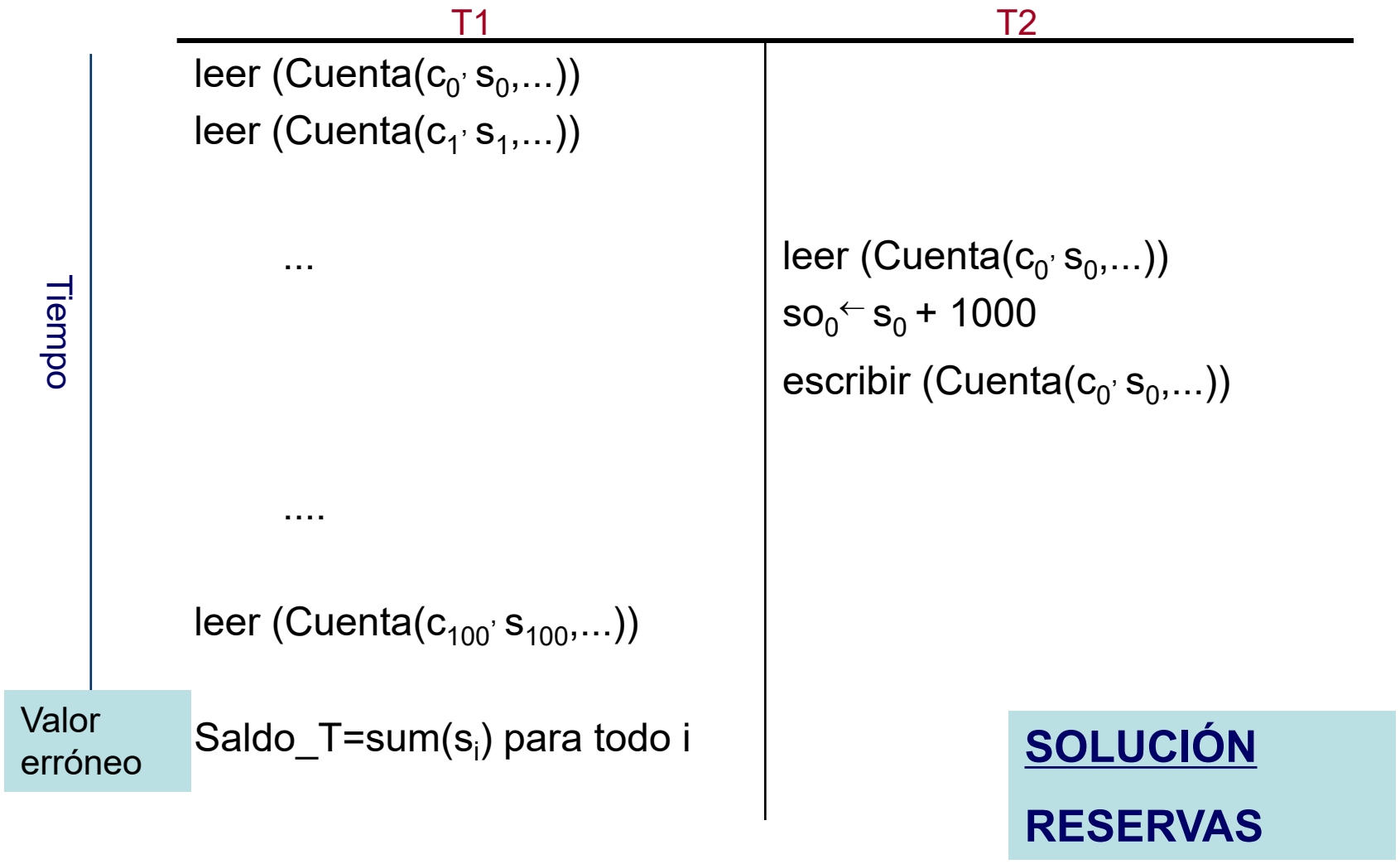


SOLUCIÓN

RESERVAS

2.3.- Control de accesos concurrentes

Información incoherente

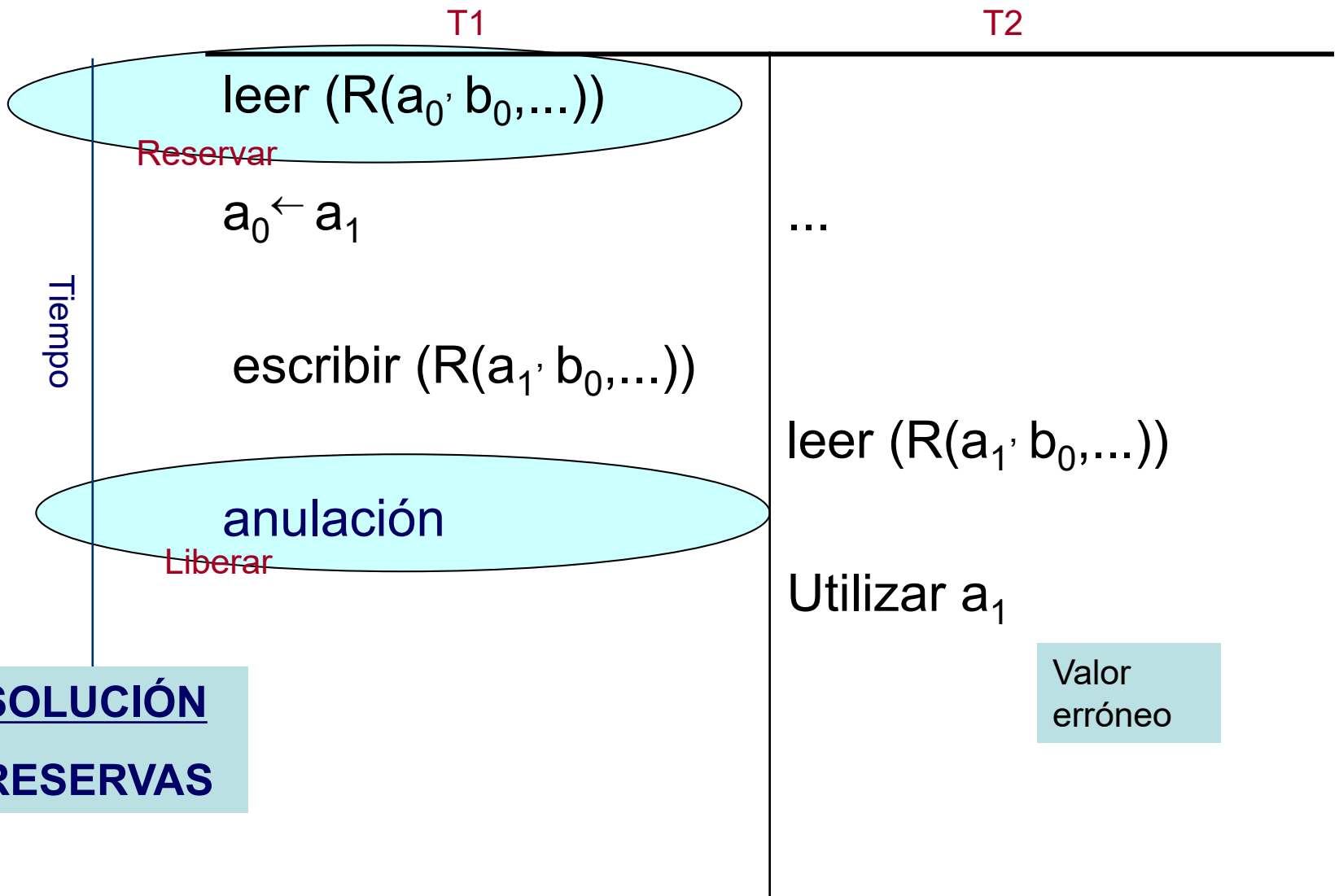


SOLUCIÓN

RESERVAS

2.3.- Control de accesos concurrentes

Lectura de datos sin confirmar



SOLUCIÓN

RESERVAS

3.- Recuperación y Seguridad

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 base de datos

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.

La pérdida de datos “confirmados” es inadmisible con la tecnología actual

3.1.- Reconstrucción de la base de datos

Visión anticuada de la reconstrucción

Copia de seguridad
del 13 de diciembre

13-12: 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

3.1.- Reconstrucción de la base de datos

Pérdidas de memoria principal

- En el espacio de tiempo entre la confirmación de una transacción y la grabación 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.

3.1.- Reconstrucción de la base de datos

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.

3.1.- Reconstrucción de la base de datos

Reconstrucción frente a fallos con pérdida de memoria

- 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).

3.1.- Reconstrucción de la base de datos

Actividades sobre el fichero diario

- Registrar las operaciones de actualización de las transacciones.
- Almacenar en disco para evitar la desaparición por un fallo del sistema.
- Grabar periódicamente a otra unidad.

3.1.- Reconstrucción de la base de datos

PROBLEMA:

- 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)

3.1.- Reconstrucción de la base de datos

Puntos de verificación: se graban en el diario periódicamente

Pasos para grabar un punto de verificación en el diario:

- 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.

Depende del SGBD

- Actualización inmediata

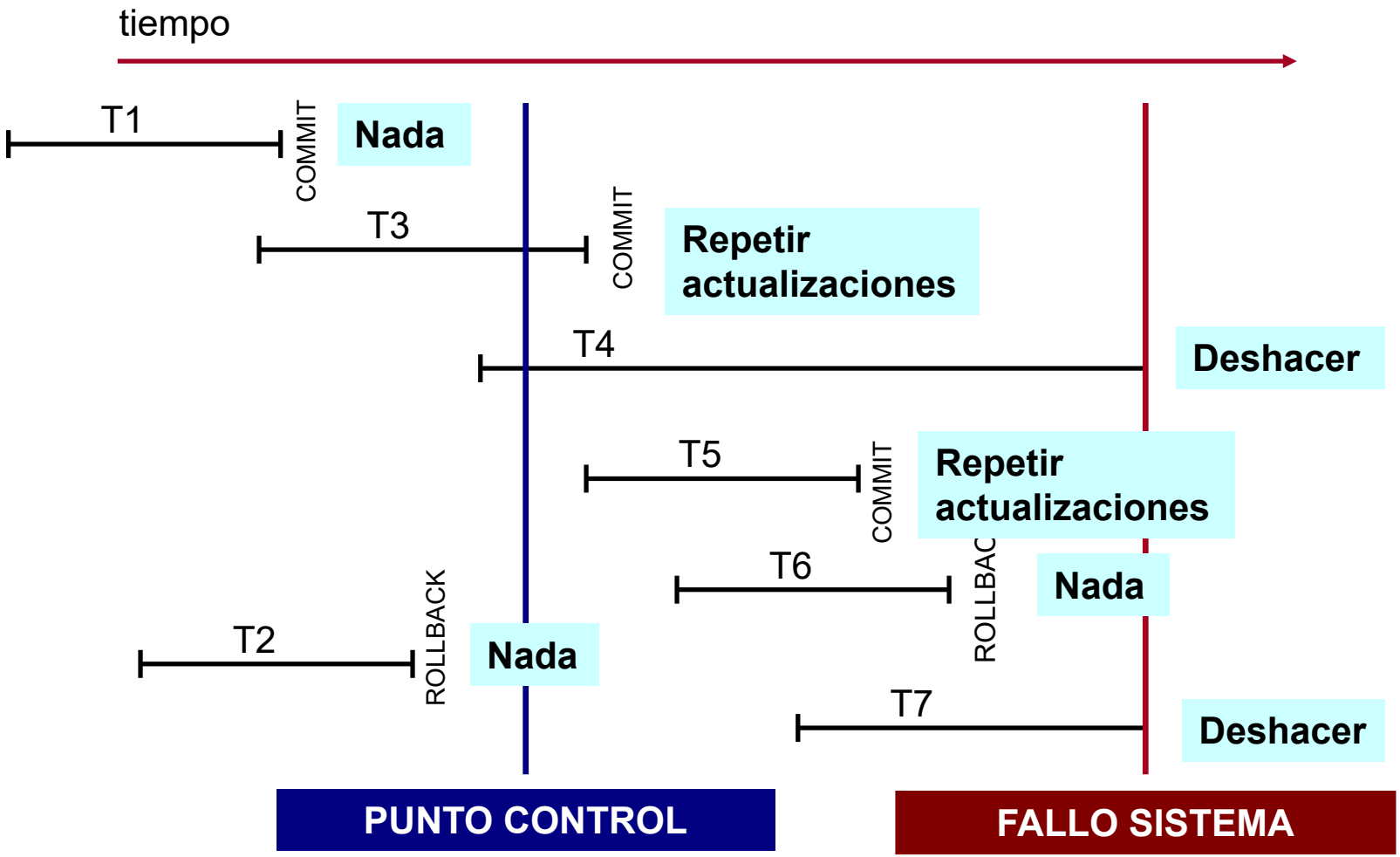
  Las actualizaciones se realizan inmediatamente en la memoria secundaria. En caso de cancelación tienen que deshacerse

- Actualización diferida

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

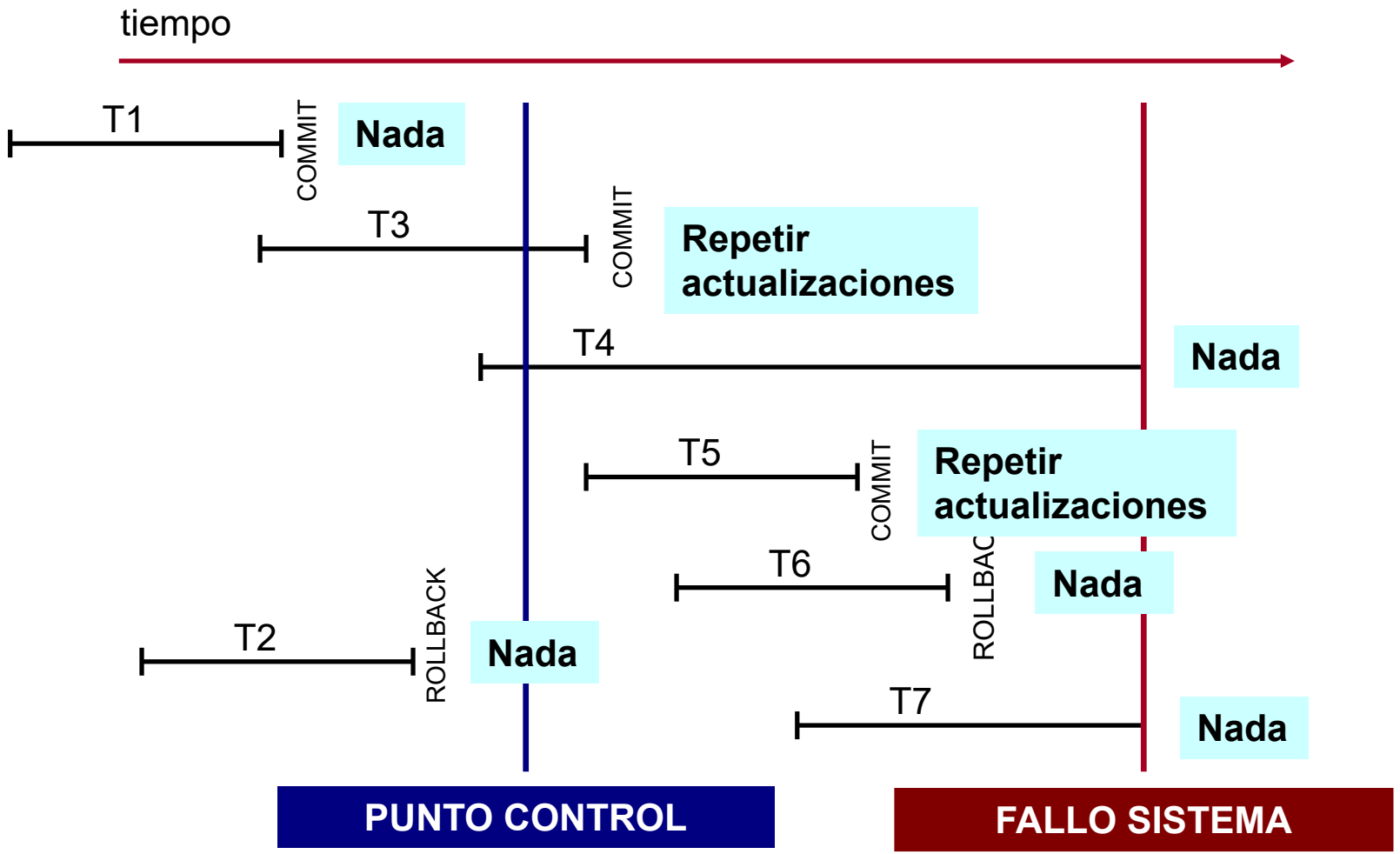
3.1.- Reconstrucción de la base de datos

Actualización inmediata



3.1.- Reconstrucción de la base de datos

Actualización diferida



3.1.- Reconstrucción de la base de datos

Reconstrucción frente a fallos 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.

3.2.- Seguridad

Objetivo:

Sólo pueden acceder a la información las personas (usuarios) autorizadas y en la forma autorizada.

3.2.- Seguridad

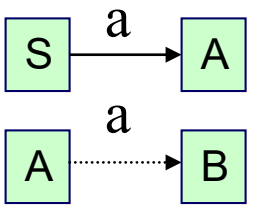
Técnicas:

- Identificación del usuario
- Determinación de los accesos permitidos:
  - Lista de autorizaciones por usuario (objeto y operaciones permitidas)
  - Niveles de autorización (menos flexible)
- Gestión de autorizaciones transferibles: traspaso de autorizaciones de un usuario a otro

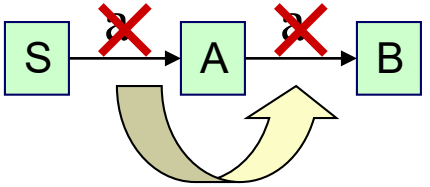
3.2.- Seguridad

Requerimientos para realizar la gestión de autorizaciones transferibles:

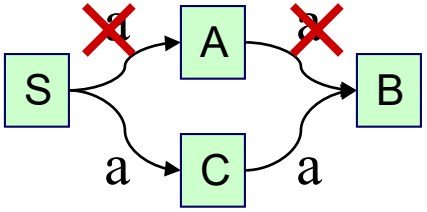
- Conocimiento de las autorizaciones de acceso de cada usuario (cuáles son transferibles a terceros y cuáles no).
- Transferencia de una autorización de un usuario a otro (en modo transferible o no).



- 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’

3.2.- Seguridad

Concesión de autorizaciones en SQL:

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]

3.2.- Seguridad

Revocación de autorizaciones en SQL:

REVOKE

{ ALL |

SELECT |

INSERT [(nom_atr$_1$,…, nom_atr$_n$)] |

DELETE |

UPDATE [(nom_atr$_1$,…, nom_atr$_n$)] }

ON nom_relación **FROM** {usuario$_1$,…, usuario$_m$ | PUBLIC}

3.2.- Seguridad

Privacidad, seguridad e implicaciones éticas y legales:

- Proteger el acceso y difusión de datos personales por usuarios no autorizados.
- Prohibir la cesión a terceros de datos 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.