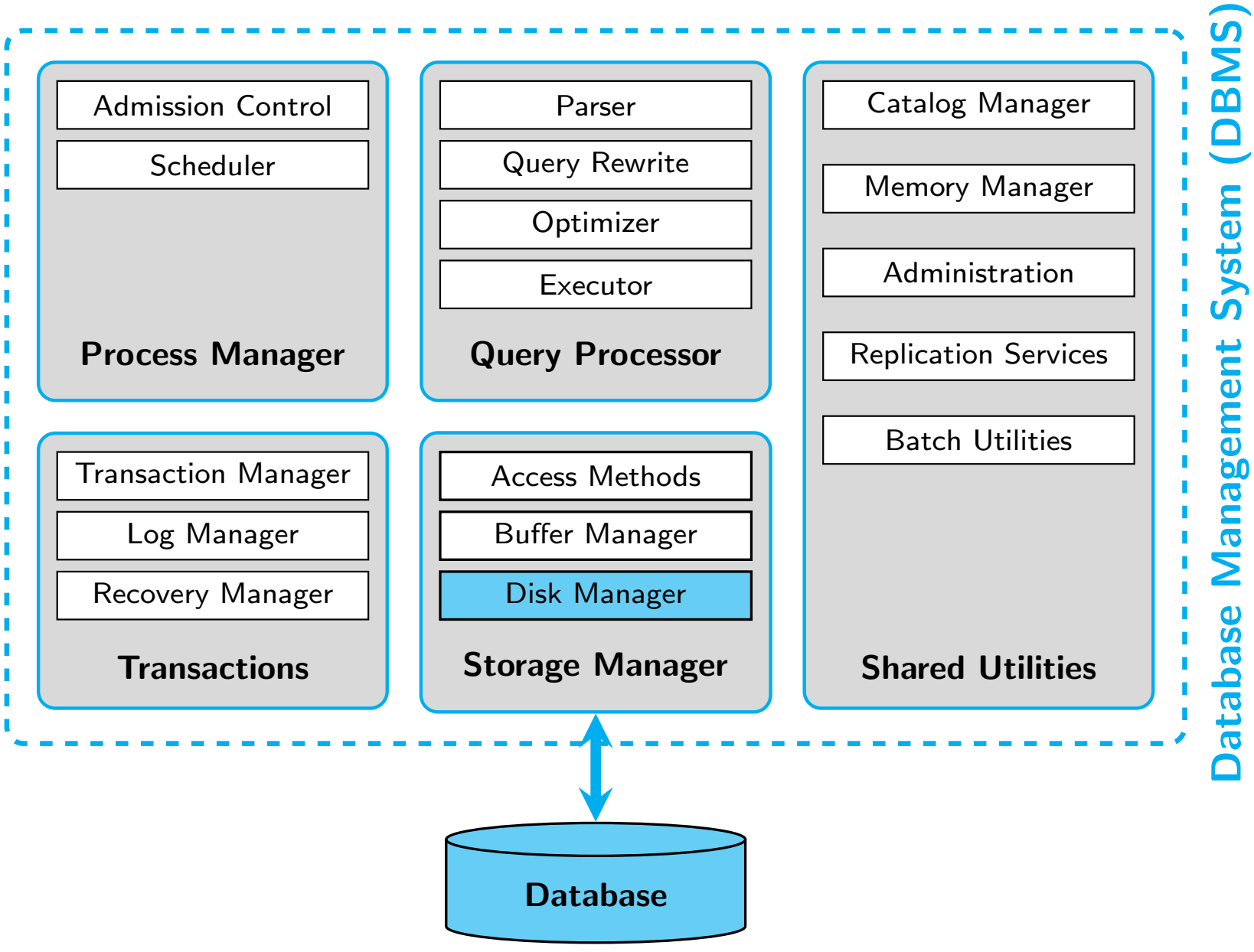
Almacenamiento de datos

Clase 02

IIC 3413

Prof. Cristian Riveros

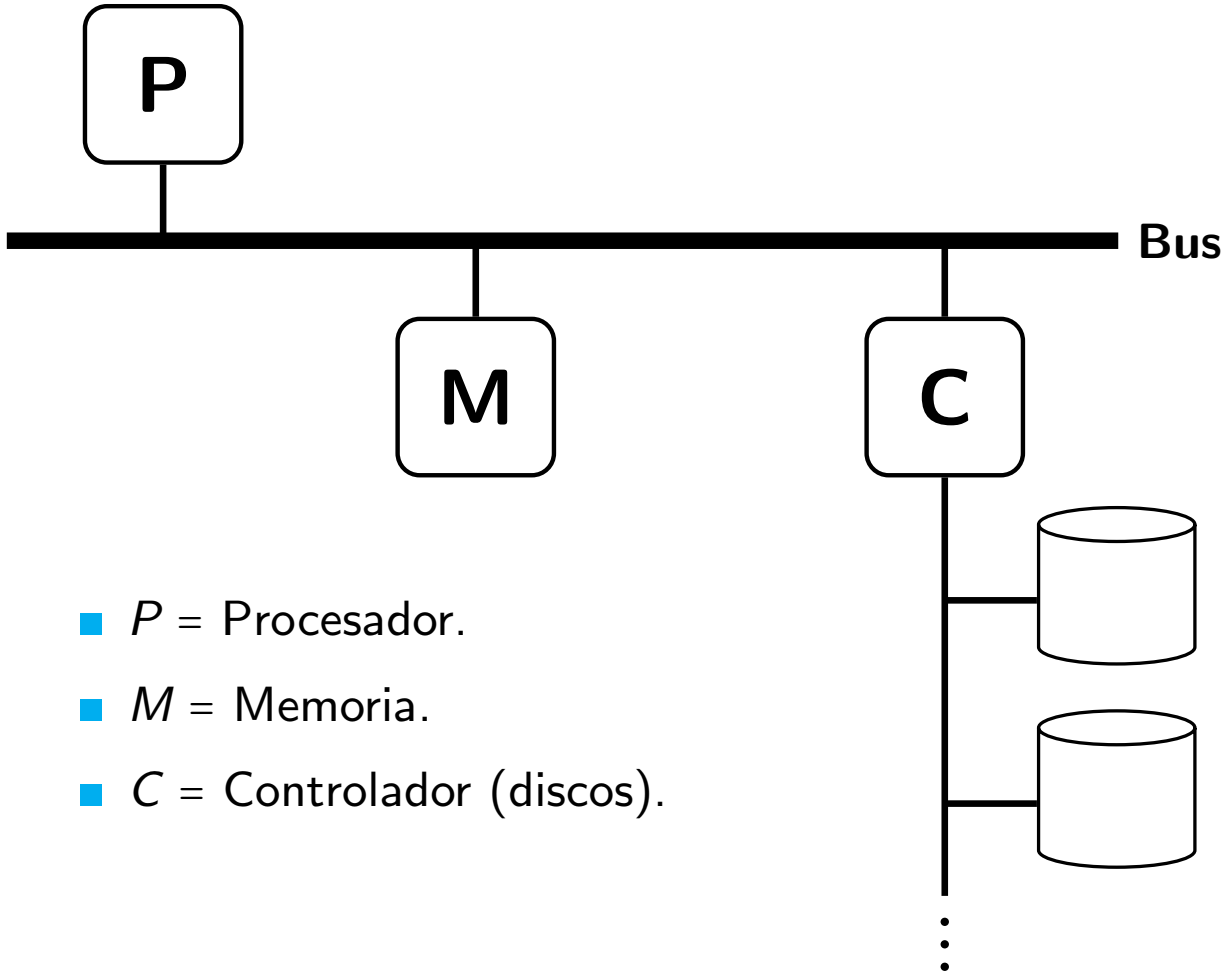
Almacenamiento de datos



Outline

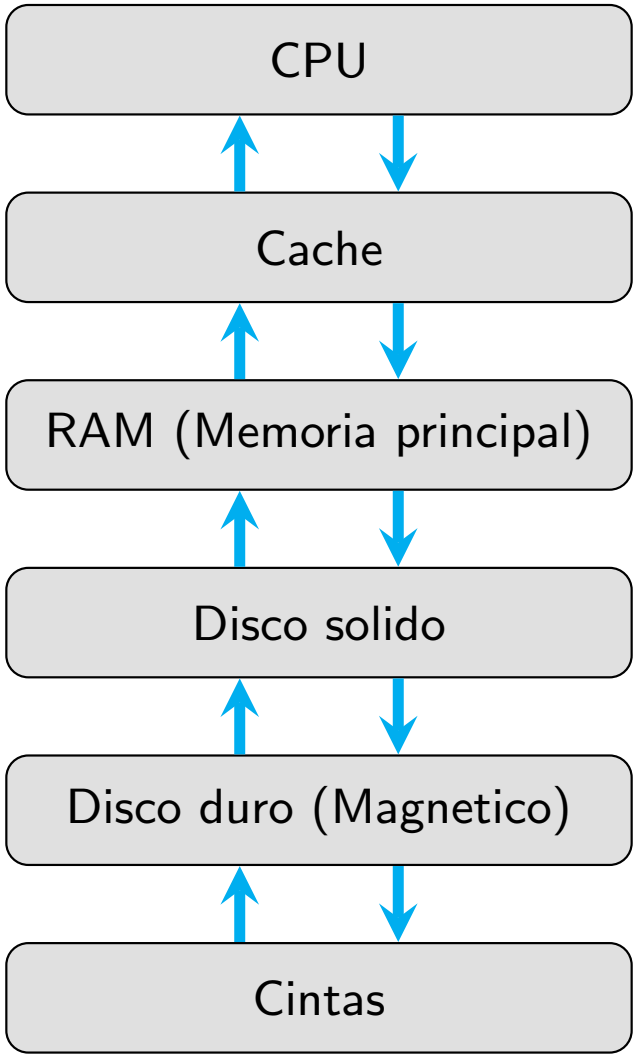
Outline

Modelo de computación y almacenamiento

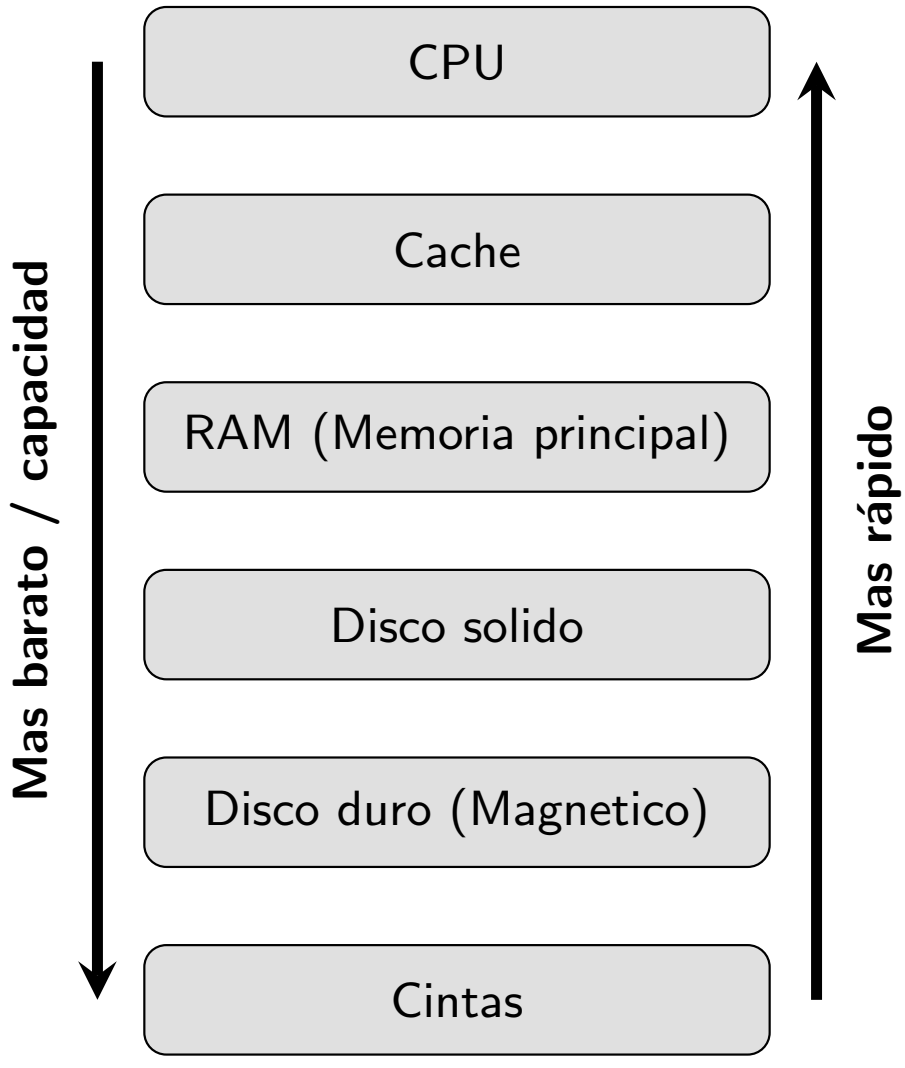


- P = Procesador.
- M = Memoria.
- C = Controlador (discos).

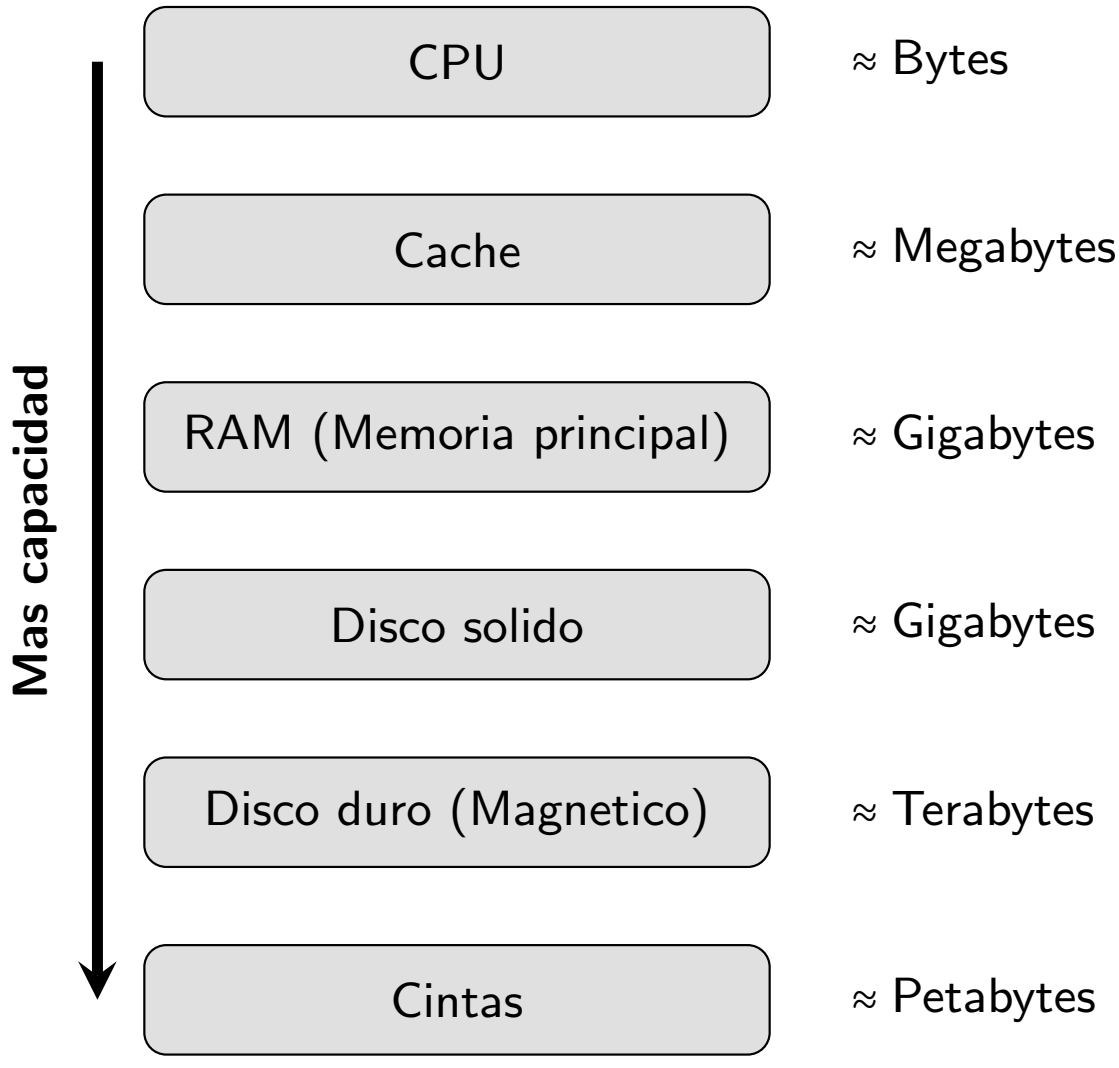
Distintos niveles de almacenamiento



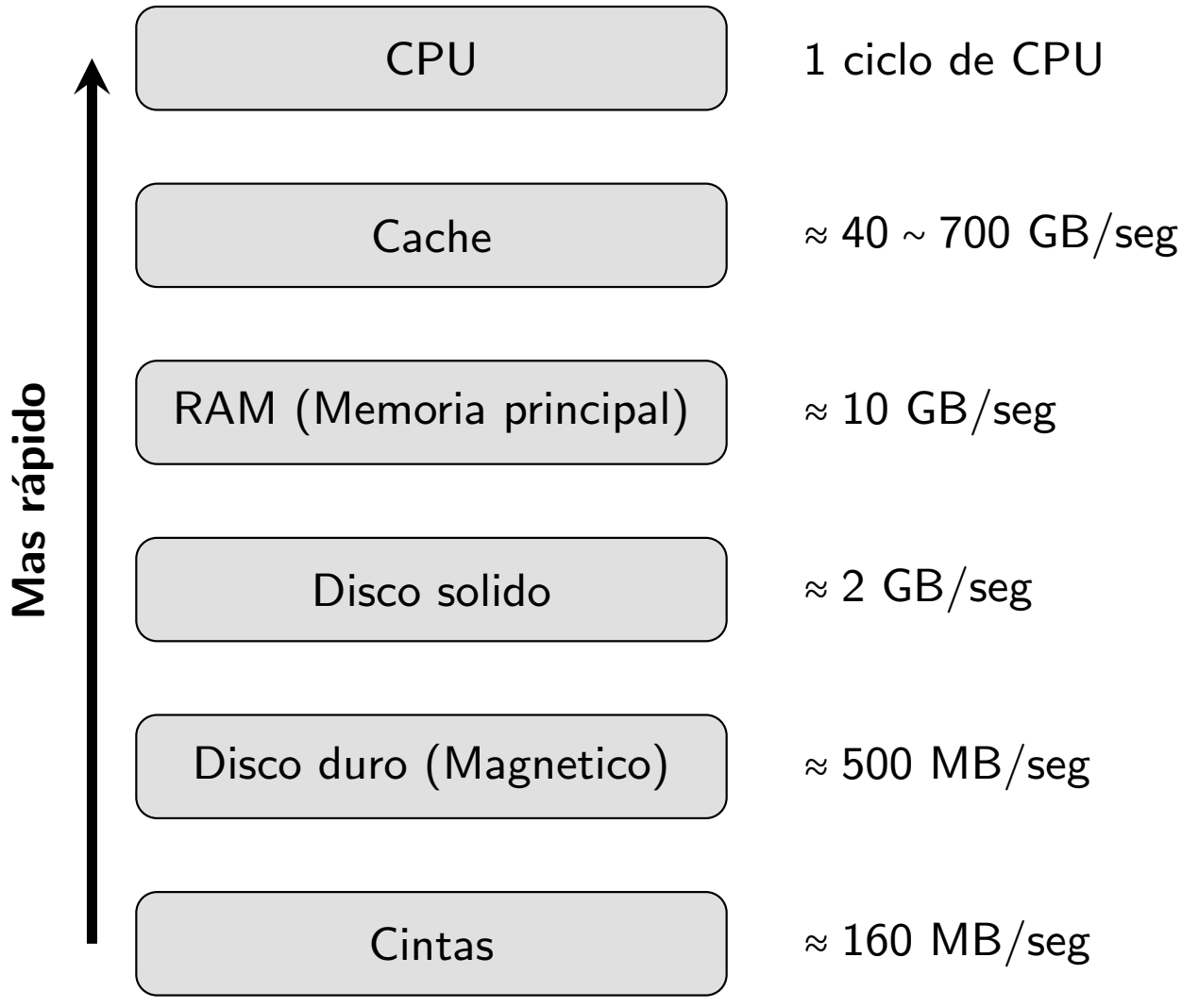
Distintos niveles de almacenamiento



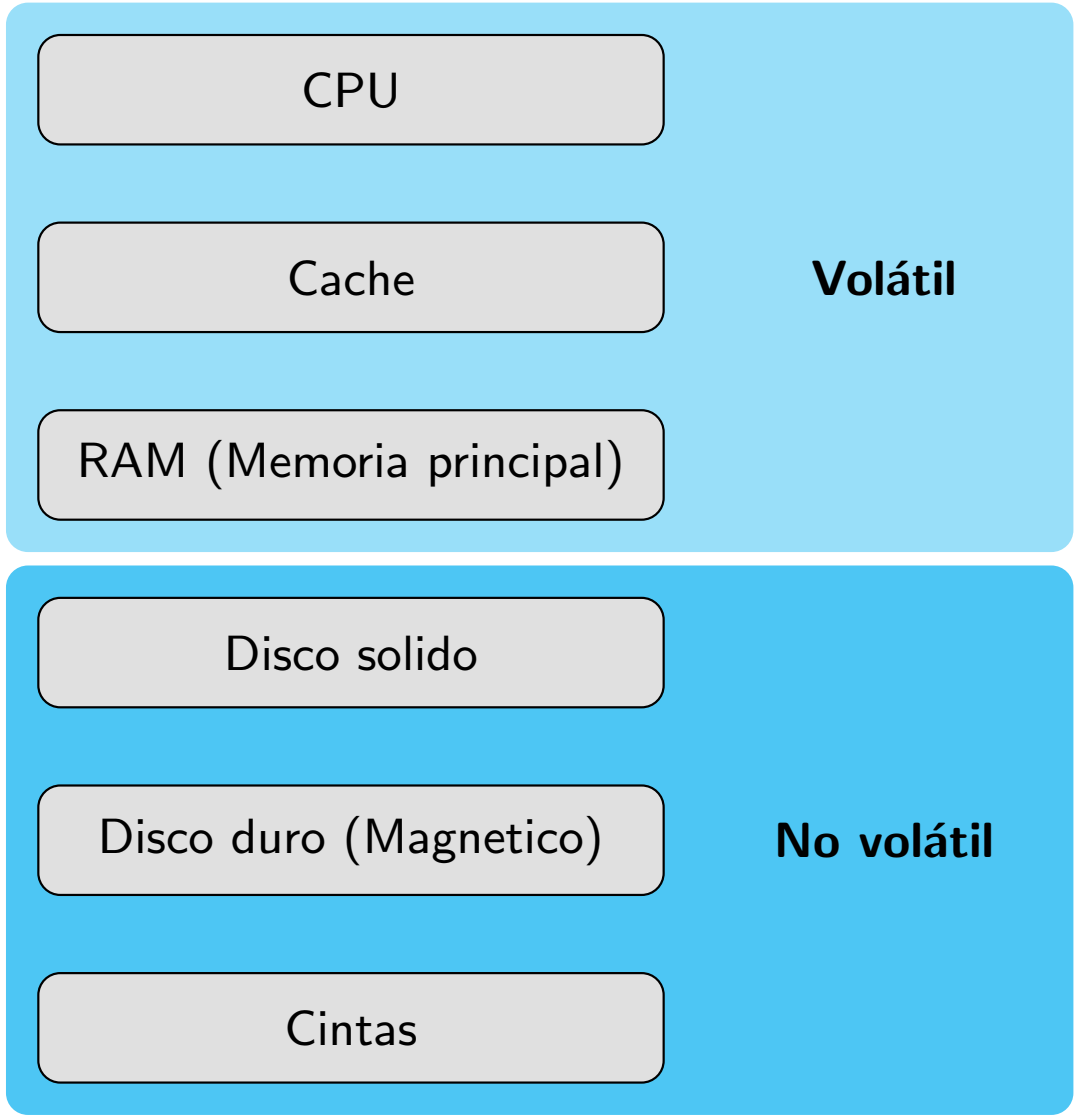
Distintos niveles de almacenamiento



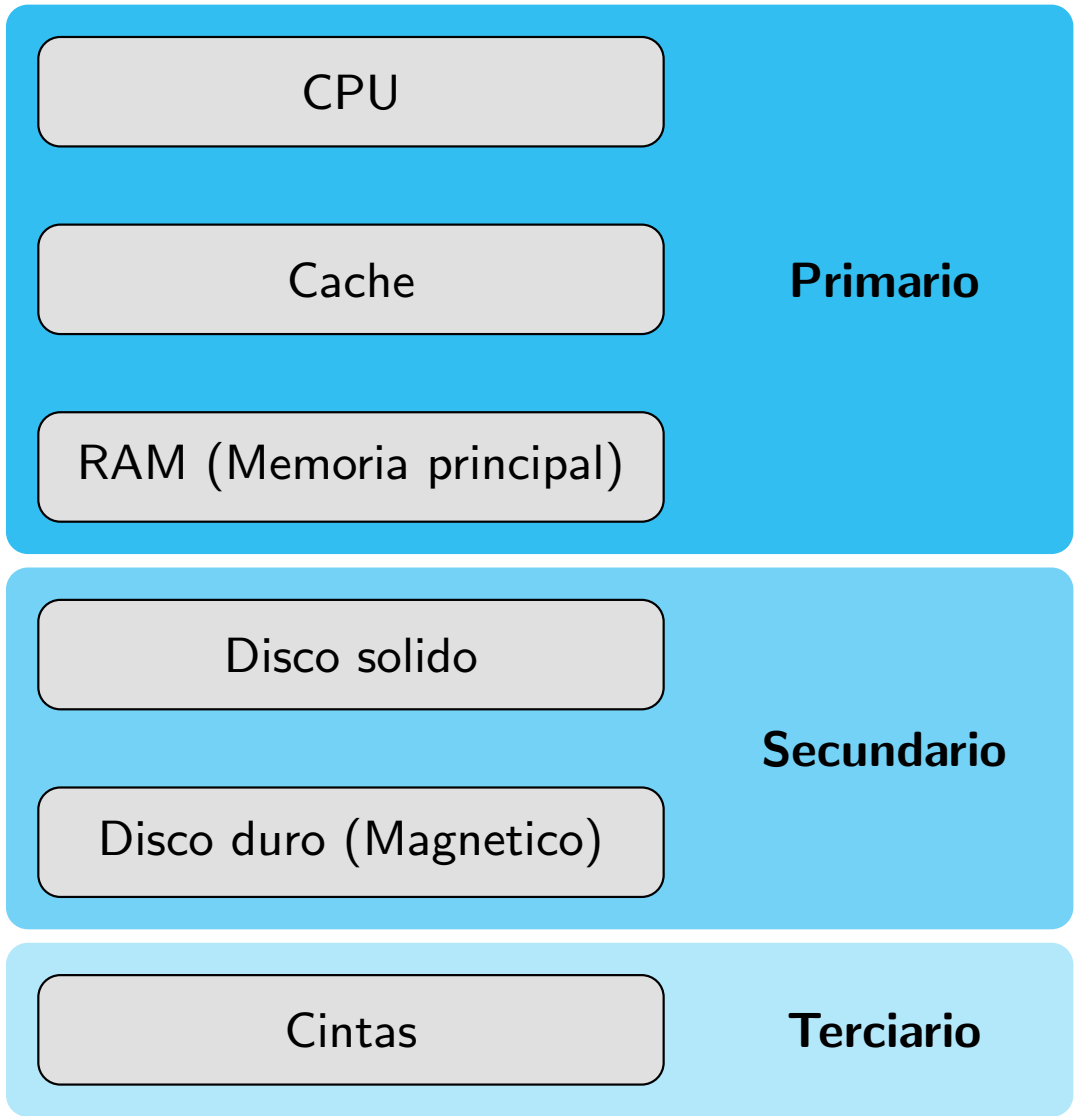
Distintos niveles de almacenamiento



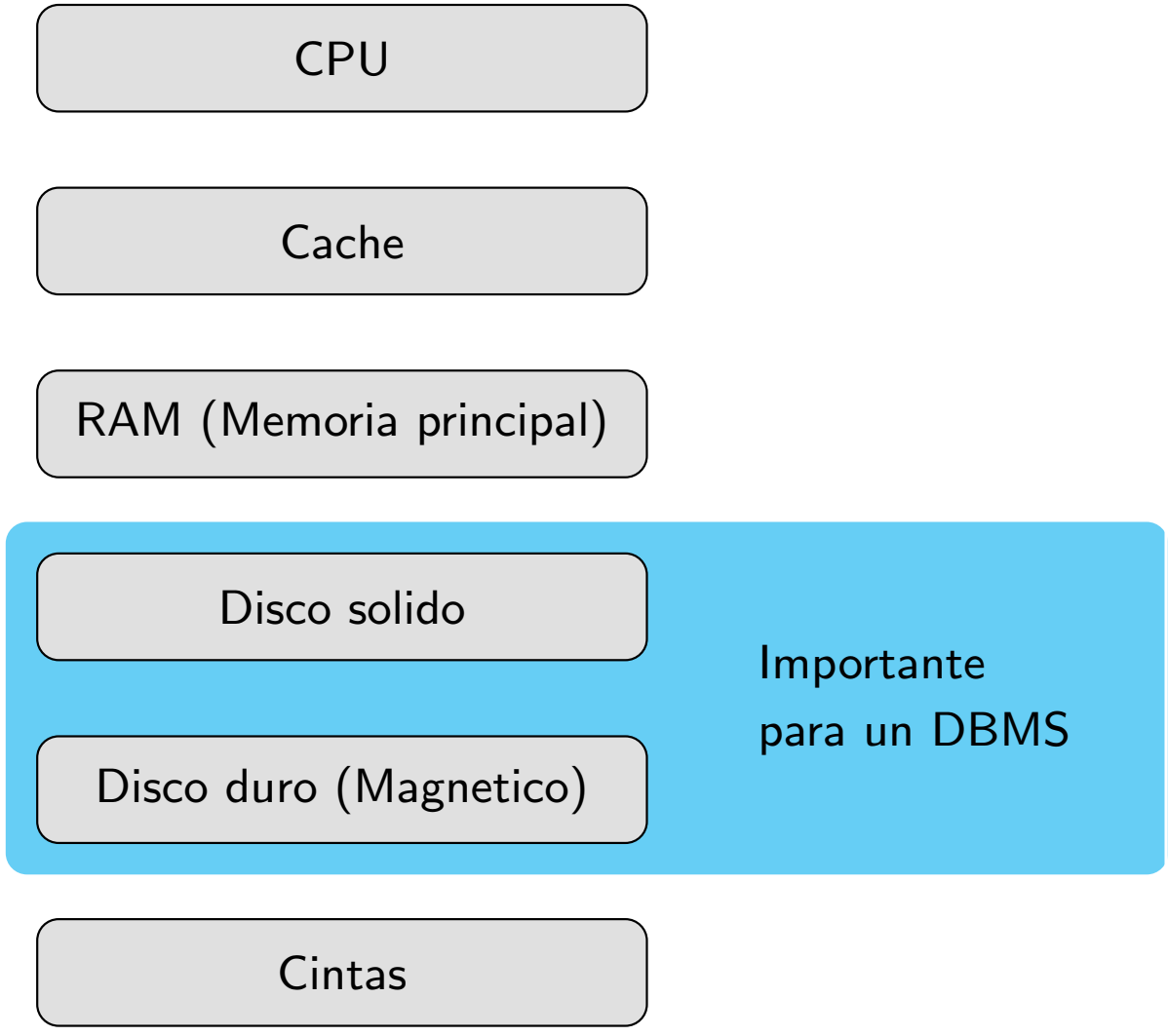
Distintos niveles de almacenamiento



Distintos niveles de almacenamiento



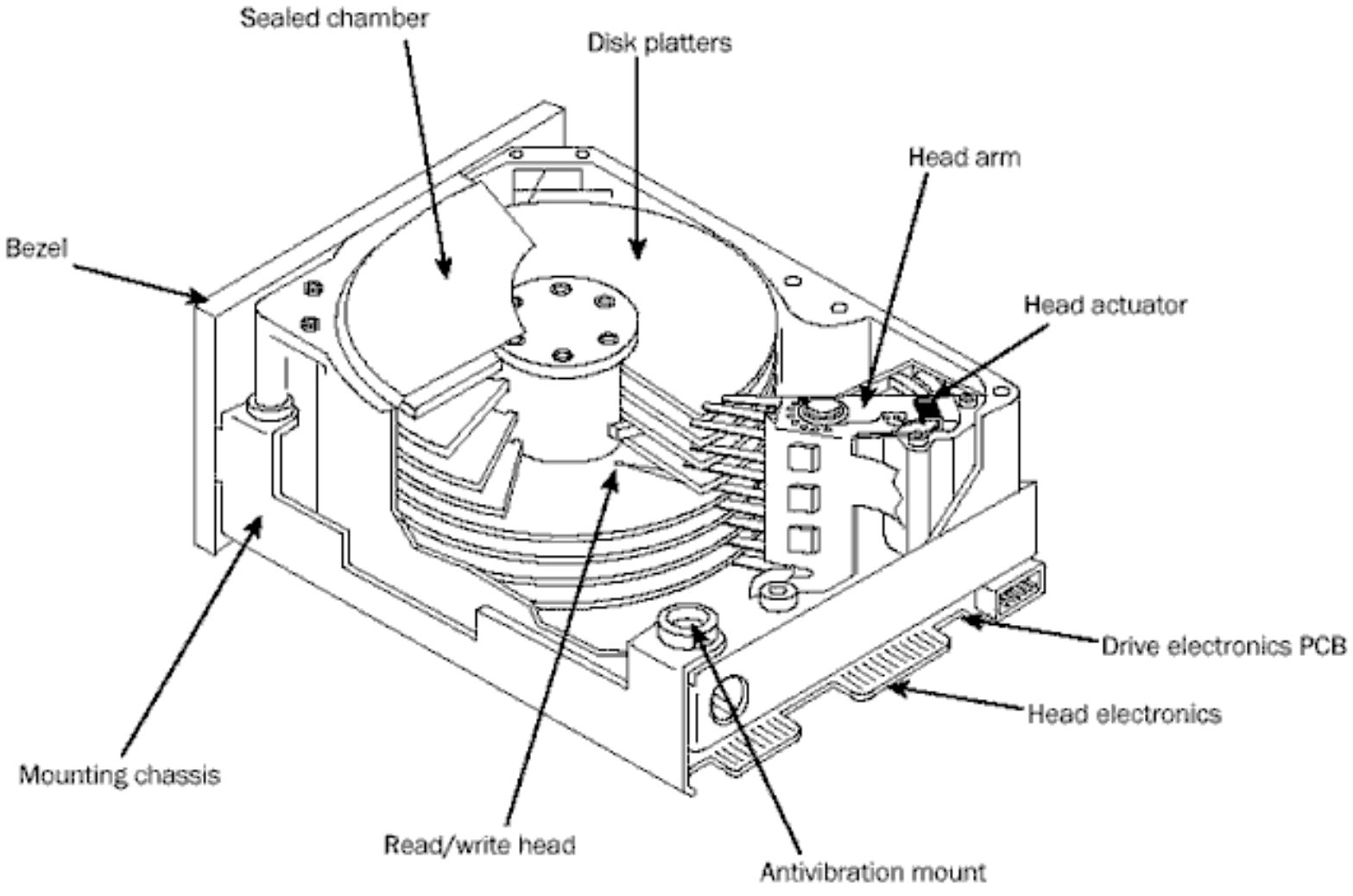
Distintos niveles de almacenamiento



Outline

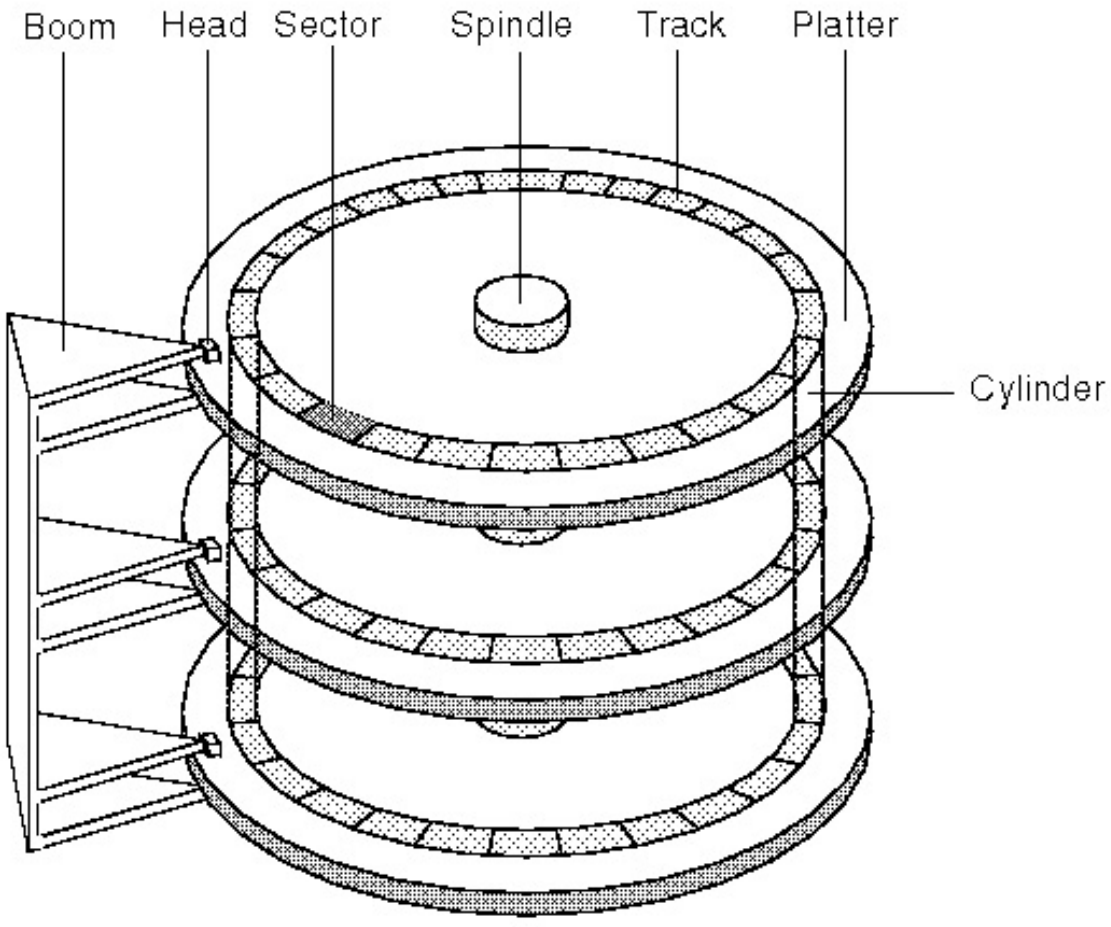
Disco duro

Principal fuente de almacenamiento secundario (no-volátil).



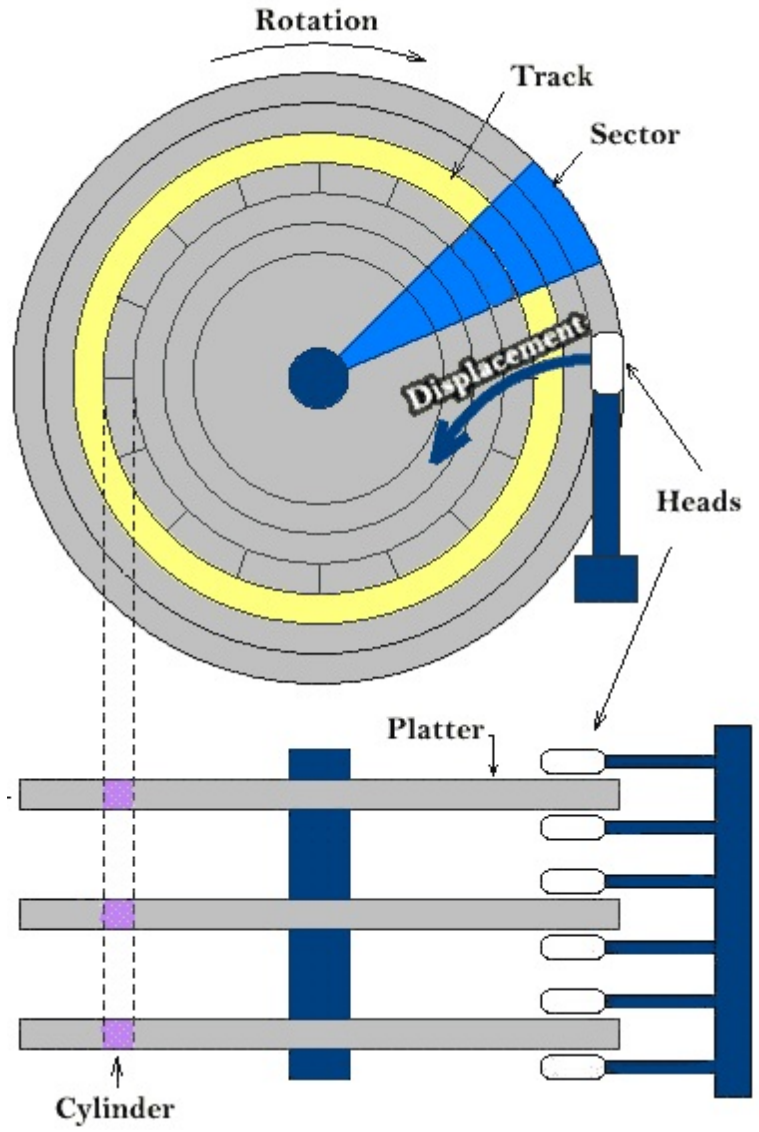
Componentes de un disco duro

- Sector
- Gap
- Track
- Cilindro
- Plato
- Cabeza lectora



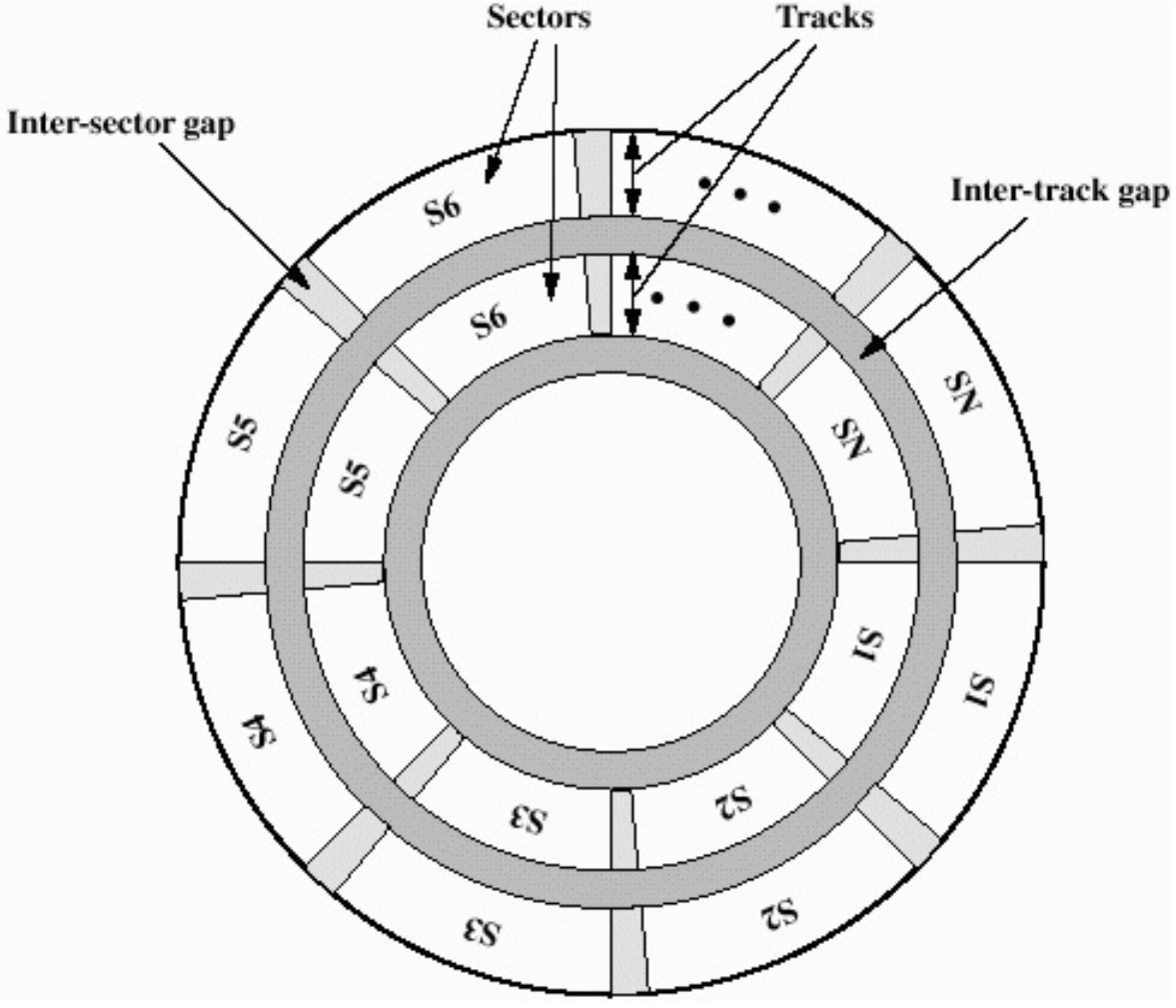
Componentes de un disco duro

- Sector
- Gap
- Track
- Cilindro
- Plato
- Cabeza lectora



Componentes de un disco duro

- Sector
- Gap
- Track
- Cilindro
- Plato
- Cabeza lectora



Componentes de un disco duro

Explicación de como funciona un disco duro.

Disco duro en acción.

Tipicas medidas de un disco duro

Superficies: 1 → 30

Diametro: 1" → 15"

- 3.5" (escritorio) y 2.5" (laptops).

Cilindros / tracks: 40 → 200.000

- ~ 100.000 tracks por pulgada.

Tamaño sectores: 512 Bytes → 50 KB

- Usualmente 512 Bytes.

Revoluciones: 4.200 rpm → 15.000 rpm

- Hoy en día: 7.200 rpm.

Ejemplo de disco duro

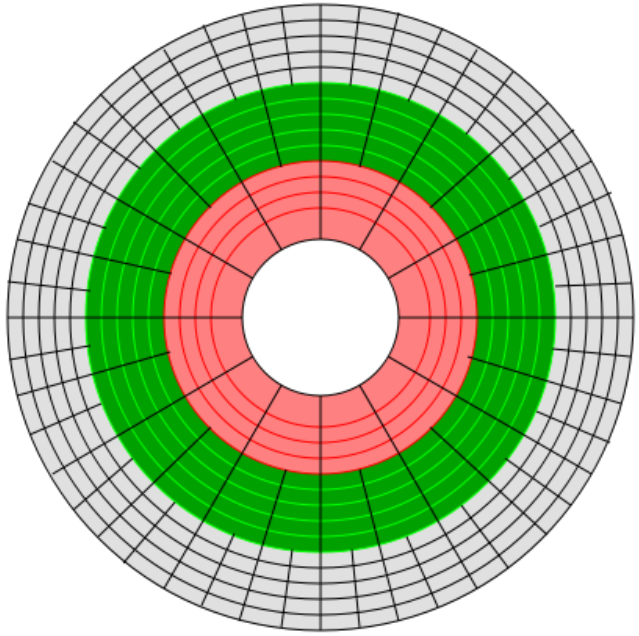
MiDiscoDuro 747 (≈ disco duro del 2008)

Superficies	:=	16	superficies (8 platos dobles)
Tracks	:=	$2^{16} = 65.536$	tracks por superficie
Sectores	:=	$2^{8} = 256$	sectores por track
$\lvert$ Sector $\rvert$	:=	$2^{12} = 4.096$	bytes por sector

- Capacidad del disco:

$$\text{Superficies} \times \text{Tracks} \times \text{Sectores} \times \#\text{Sector} \approx 1\ TB$$

Zone bit recording



- Tracks divididos por zona de distinta densidad de sectores.
- Datos van desde afuera hacia adentro.
- Transfer rate externo mas rápido que interno.

Sectores vs Blocks vs Pages

Definición

- Sector: unidad **fisica** minima de almacenamiento (**Disco**).
- Block: unidad **lógica** minima de almacenamiento (**Disco**).
- Page: unidad **lógica** minima de almacenamiento (**Sistema**).

Demora en la busqueda de los datos

Tiempo total	=	Tiempo de busquedad	(Seek time)
		+	
		Retraso rotacional	(Rotational delay)
		+	
		Velocidad de transferencia	(Transfer rate)

Definición

- Tiempo de busqueda: tiempo en posicionar cabeza en el track.
- Retraso rotacional: tiempo en rotar el disco hasta el primer sector.
- Vel. de transferencia: tiempo en leer los sectores (incl. gaps).

Demora en la busqueda de los datos

Tiempo total	=	Tiempo de busquedad	(Seek time)
		+	
		Retraso rotacional	(Rotational delay)
		+	
		Velocidad de transferencia	(Transfer rate)

Definición

- Tiempo de busqueda: 3 ms $\rightarrow$ 15 ms (promedio).
- Retraso rotacional: 2 ms $\rightarrow$ 7 ms (promedio).
- Vel. de transferencia: $\rightsquigarrow$ 0 (dependiendo del tamaño de los datos)

Ejemplo de tiempos de demora

MiDiscoDuro 747

- Rotación: 7200 rpm (8.33 ms por vuelta).
- Movimiento brazo:

  1 ms para empezar/parar brazo.
  1 ms por cada 4000 tracks.

  Numero de tracks: 65.536 tracks.
- Transferencia de datos: 0.13 ms por bloque.

	Min	Max	Promedio
Seek	0 ms	17.38 ms	6.46 ms
Rotational	0 ms	8.33 ms	4.17 ms
Transfer	0.13 ms	0.13 ms	0.13 ms
Total	0.13 ms	25.84 ms	10.76 ms

¿cuánto es la demora si debo buscar 1000 bloques distintos?

Escritura de bloques

Costo de escritura similar al costo de leer un bloque (o peor).

Para modificar, uno debe:

- Leer el bloque.
- Modificar el bloque en memoria.
- Escribir el bloque.

Moraleja (sobre uso de un disco duro)

1. Tiempos de busqueda en un disco duro pueden ser significativos.
2. Datos relacionados deben ser almacenados en bloques contiguos.
3. Minimizar número de accesos a disco duro.

Modelo de computación basado en I/O

Hechos / suposiciones:

- Espacio del disco duro >> espacio en RAM.
- Tiempo de acceso a disco >> tiempo de acceso RAM.

Acceso disco duro:	$\approx$ 10 ms
Acesso RAM:	$<$ 0.1 ms

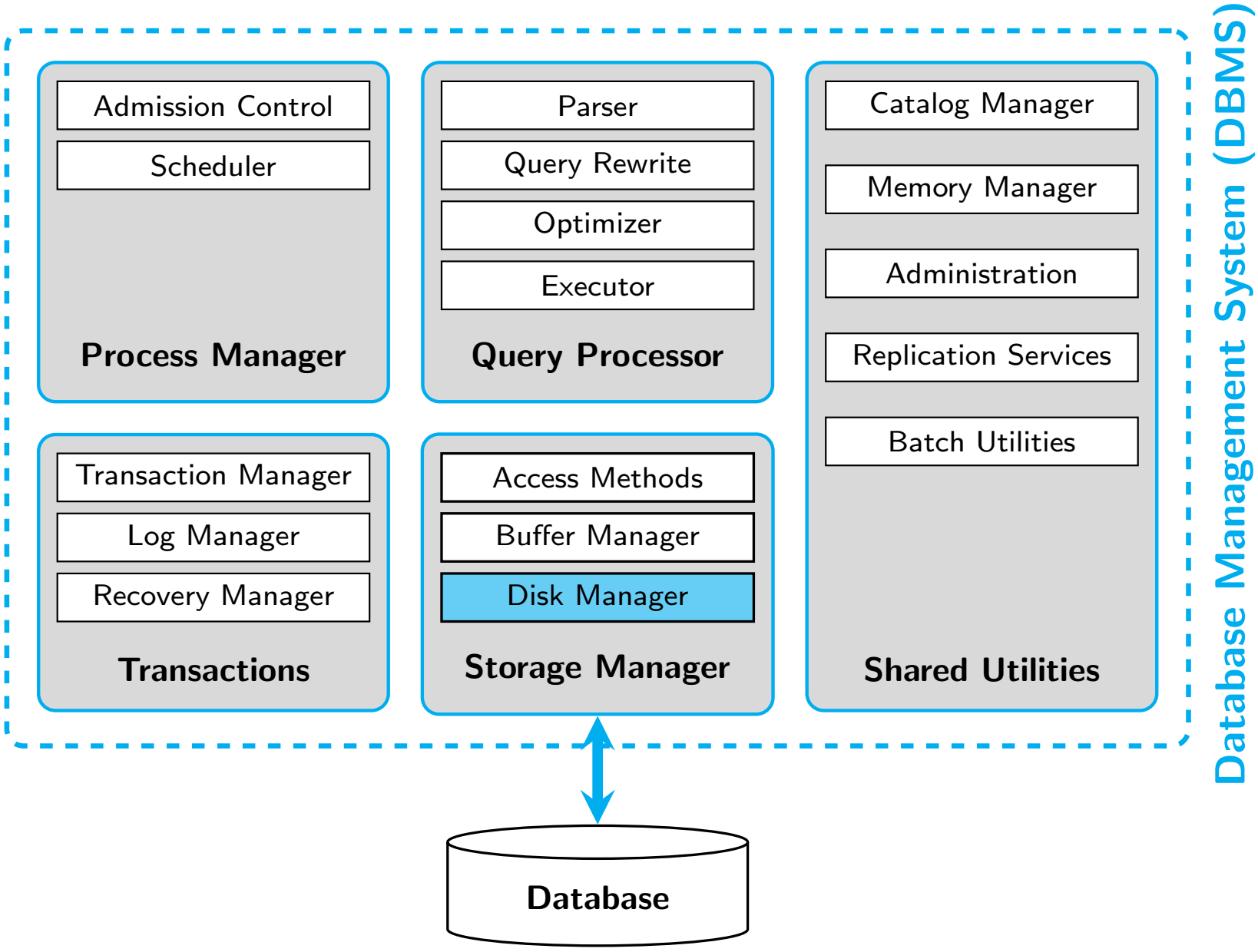
Dominación del costo I/O:

> *"El tiempo para acceder al disco es mucho mayor que el tiempo necesario para manejar datos en memoria. Entonces, el número de accesos a disco (Disk I/O) es una buena aproximación al tiempo del algoritmo y, por lo tanto, debe ser minimizado."*
>
> Garcia-Molina, Ullman, & Widom, 2008

Esto es lo más importante de esta clase

Outline

Optimizaciones para manejo del almac. secundario



Optimizaciones para manejo del almac. secundario

1. Localidad de los datos.
2. Pre-fetch.
3. Planificación del disco.
4. Multiples discos.
5. Multiples copias (mirroring).

1. Localidad de los datos

Principio:

Si las tuplas t y t' son parte de la misma relación R, entonces almacenar t:

- en el **mismo bloque** que t', o, si no,
- en el **mismo track** que t', o, si no,
- en el **mismo cilindro** que t', o, si no,
- en el **cilindro contiguo** a t'.

2. Pre-fetch

Principio:

- Adivinar y traer bloques serán solicitados en el futuro.
- Para una solicitud de un bloque k, traer también bloques contiguos

$$k+1, k+2, \ldots, k+N$$

(N es una cantidad fijada por el usuario).

Ejemplo

- Acceso a una tupla de una relación.
- Acceso a una columna (Column store, ver mas adelante).
- Acceso a la estructura de un índice (ver clase de índices).

3. Planificación (o scheduling) del disco

Principio:

Dada una secuencia de solicitudes $o_1, \ldots, o_n$ al disco, reordenar el orden de las solicitudes a $o_{i_1}, \ldots, o_{i_n}$ de tal forma de minimizar el tiempo total.

Ejemplo (Algoritmo del ascensor)

- Disco es como un ascensor, donde sus pisos son los cilindros del disco.
- Brazo del disco mantiene su dirección si quedan solicitudes por delante.

Cilindro solicitado	Llegada solicitud	Tiempo Elevador	Tiempo FIFO
8000	0 ms	8 ms	8 ms
24000	0 ms	24 ms	24 ms
36000	0 ms	36 ms	36 ms
5000	10 ms	87 ms	67 ms
46000	20 ms	46 ms	108 ms
30000	30 ms	62 ms	124 ms

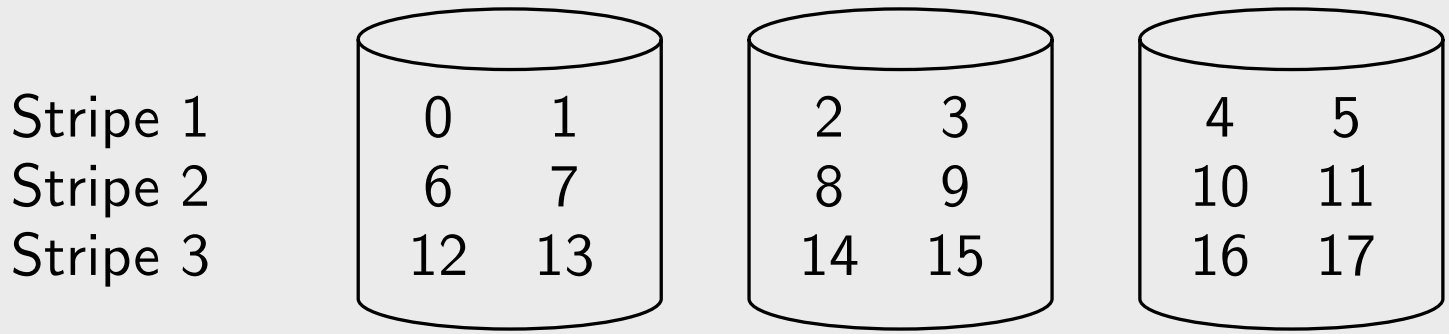
Considere velocidad del brazo de 1000 tracks por ms.

4. Multiples discos

Principio:

Distribuir tuplas en multiples discos para asi tener acceso concurrente a las relaciones y tuplas.

Ejemplo (Striping)



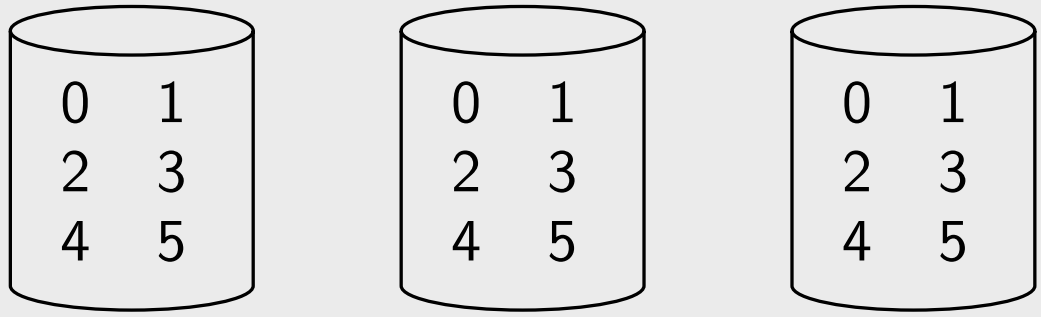
5. Multiples copias

Principio:

Mantener multiples copias de los datos en distintos discos para:

- Disponibilidad de los datos en caso de fallas.
- Mayor velocidad al leer datos (pero menor velocidad al escribir).

Ejemplo (Mirroring)



Comparación de optimizaciones

Dos casos importantes:

A. Procesamiento *Batch* (OLAP = On-Line Analytical Processing)

- Consultas que no requieren una respuesta instantánea.
- Un solo proceso usando el disco.
- Accesos al disco pueden ser predecidos.

B. Procesamiento Online (OLTP = On-Line Transaction Processing)

- Consultas requieren una respuesta instantánea.
- Varios procesos accediendo el disco.
- Poca capacidad de predicción de accesos al disco.

Comparación de optimizaciones

Localidad de los datos

- Bueno para A, no tan útil para B.

Pre-fetch

- Bueno para A, poca ayuda para B.

Planificación del disco

- Bueno para B, especialmente cuando hay muchas solicitudes.

Multiples discos y multiples copias

- Bueno para ambos, especialmente para B.

Outline

Nuevas tecnologías

- Discos de estado solido.
- Almacenamiento en la red.
- Almacenamiento en la nube.

Discos de estado solido (SSD)

Ventajas / desventajas:

- Misma interfaz I/O.
- No hay demora "mecánica" ni espacial.
- Asimetría entre escritura y lectura.
- "Borrar antes de escribir".
- Mejores propiedades físicas y energéticas.
- Mucho mas costoso ($$$).

Es necesario repensar los sistemas de BD relacionales.

Discos de estado solido (SSD)