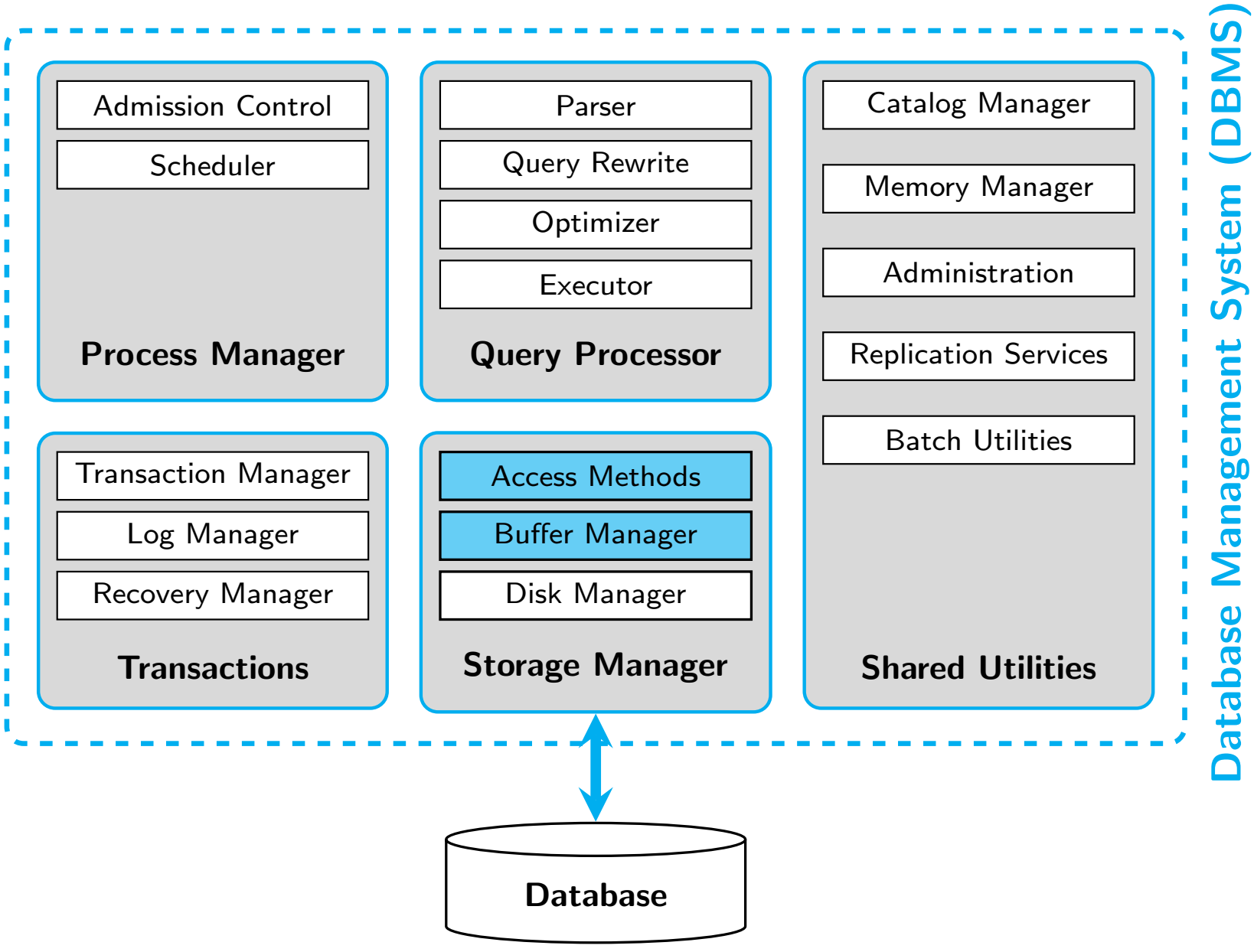
Buffer manager e índices

Clase 04

IIC 3413

Prof. Cristian Riveros

Para esta clase



Outline

Buffer Manager

Índices

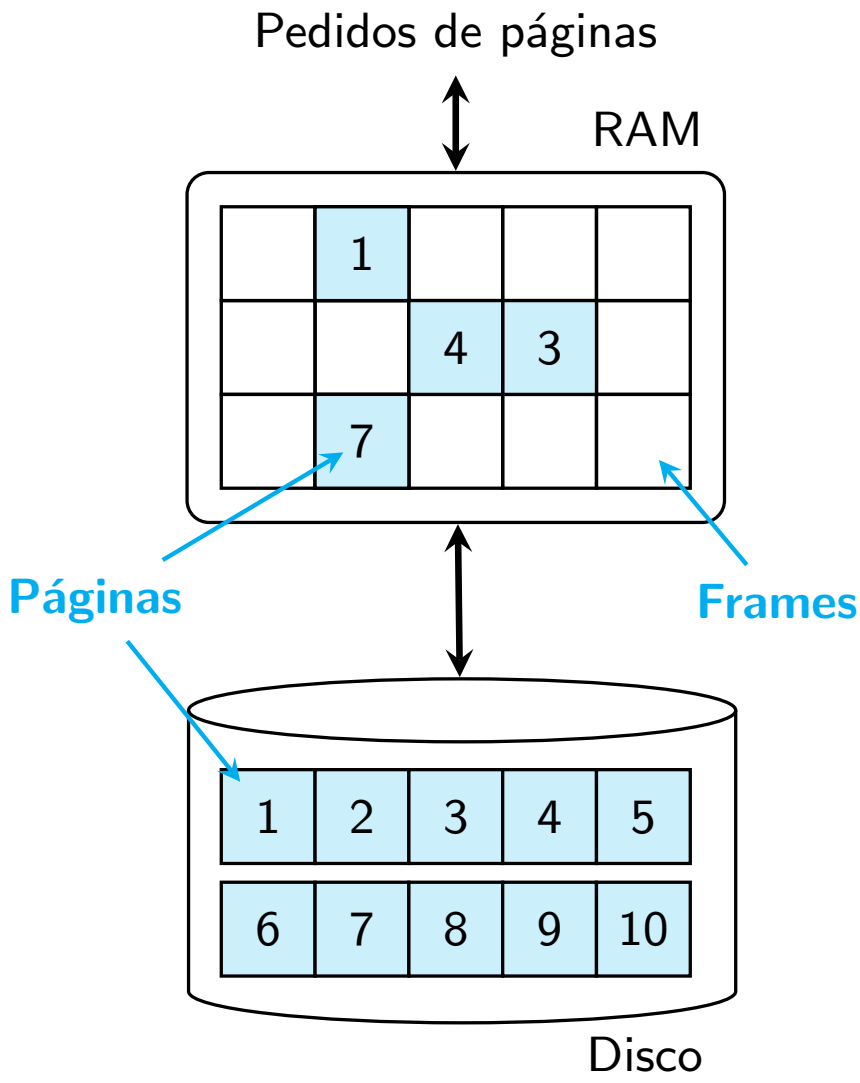
Outline

Buffer Manager

Índices

Buffer manager

- Mediador entre el disco y la memoria principal.
- Cuenta con una cantidad restringida de memoria RAM.
- Páginas son traidas a memoria a pedido.
- Encargado de decir que páginas eliminar cuando el buffer esta lleno.



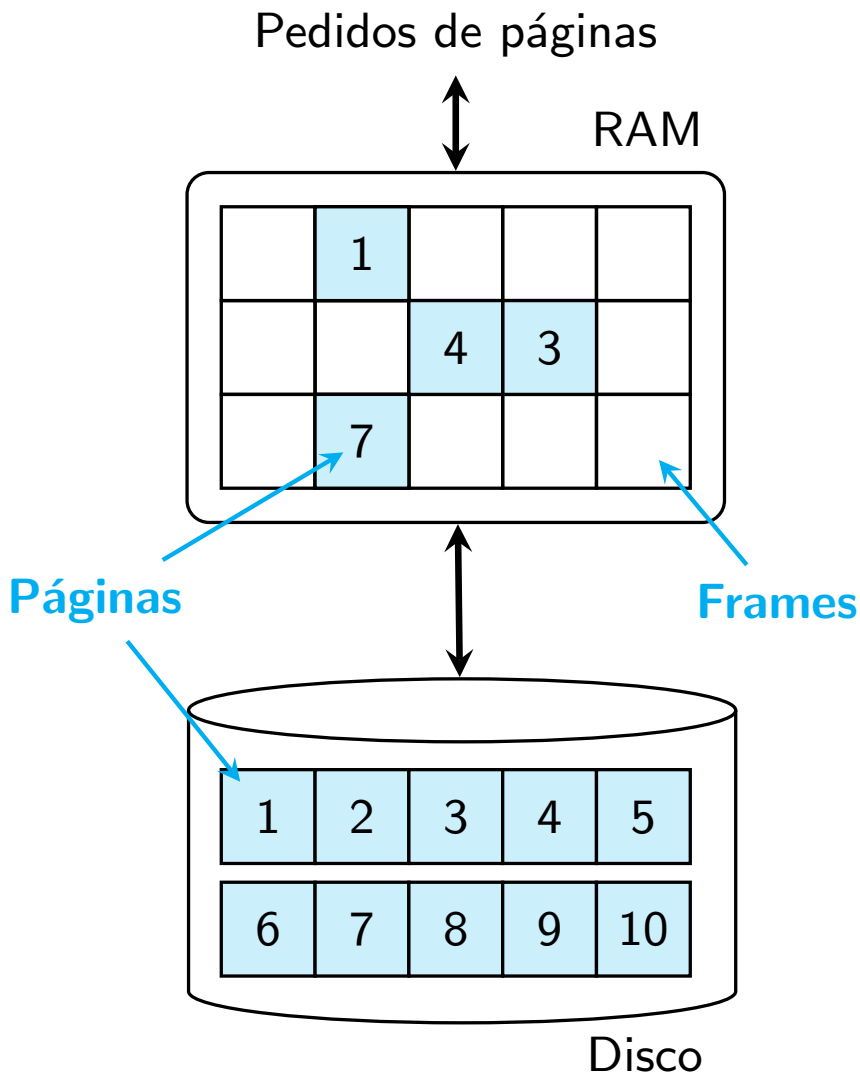
Buffer manager

Cada **frame** tiene dos variable:

1. #**pin** = cantidad de procesos que estan usando esta página.
2. **Dirty** = si el contenido es necesario guardarlo en memoria.

Procesos acceden al Buffer manager usando las funciones:

```
pin(pageno) y
unpin(pageno, dirty)
```



Interfaz para acceder el buffer manager

Función `pin( pageno )`

1. Solicita la página `pageno` al buffer manager.
2. Si la página `pageno` no esta en memoria:
   - Se selecciona un frame vació X.
   - Se trae la página `pageno` a memoria y se carga en X.
   - Se actualiza $X.\#pin := 1$ y $X.dirty := false$.
3. Si la página `pageno` esta en memoria, se incrementa su #-pin en 1.
4. Buffer manager retonar una referencia al frame que contiene `pageno`.

¿Problemas?

- ¿qué ocurre si no hay un frame vacío?
- ¿qué ocurre si todos los frames tienen #pin mayor que 1?

Interfaz para acceder el buffer manager

Función `unpin( pageno, dirty )`

1. Solicita la liberación de la página `pageno` (almacenada en el frame X).
2. Se decrementa el $X.\#pin$ en uno.
3. Se actualiza $X.dirty := true$ si la página fue modificada.

¿Problemas?

- ¿cuándo son guardados los datos de la página en disco?
- ¿qué ocurre si un proceso nunca hace unpin de su página?

Requisitos de la interfaz `pin` y `unpin`

1. Cada proceso (de una BD) debe mantener las funciones `pin` y `unpin` **correctamente anidadas**.

   $d \leftarrow$ `pin`(p);
   $\vdots$
   (proceso lee y usa los datos en la dirrección de memoria d)
   $\vdots$
   `unpin`(p, false);

2. Cada proceso (de una BD) debe liberar una página lo antes posible.

Escritura concurrente de una página

Suponga lo siguiente:

- La misma página p es solicitada por más de un proceso ($\#pin > 0$).
- Mas de un proceso modifica el mismo dato de p.

¿qué hacemos?

... estos conflictos son manejados por el **administrador de transacciones**.

(lo veremos más adelante)

Políticas de reemplazo (replacement policies)

La efectividad del buffer manager depende directamente de la política de reemplazo usada.

Diferentes tipos de póliticas o (heurísticas).

- FIFO.
- Least Recently Used (LRU).
- Clock.
- Random.

Recordar: estas son heuristicas y pueden fallar.

Buffer manager en la práctica

Varias técnicas adicionales:

- Prefetching.
- Page fixing / hating.
- Buffer particionado.

Base de Datos vs. Sistemas Operativos

¿cuáles son las ventajas de usar un buffer manager propio?

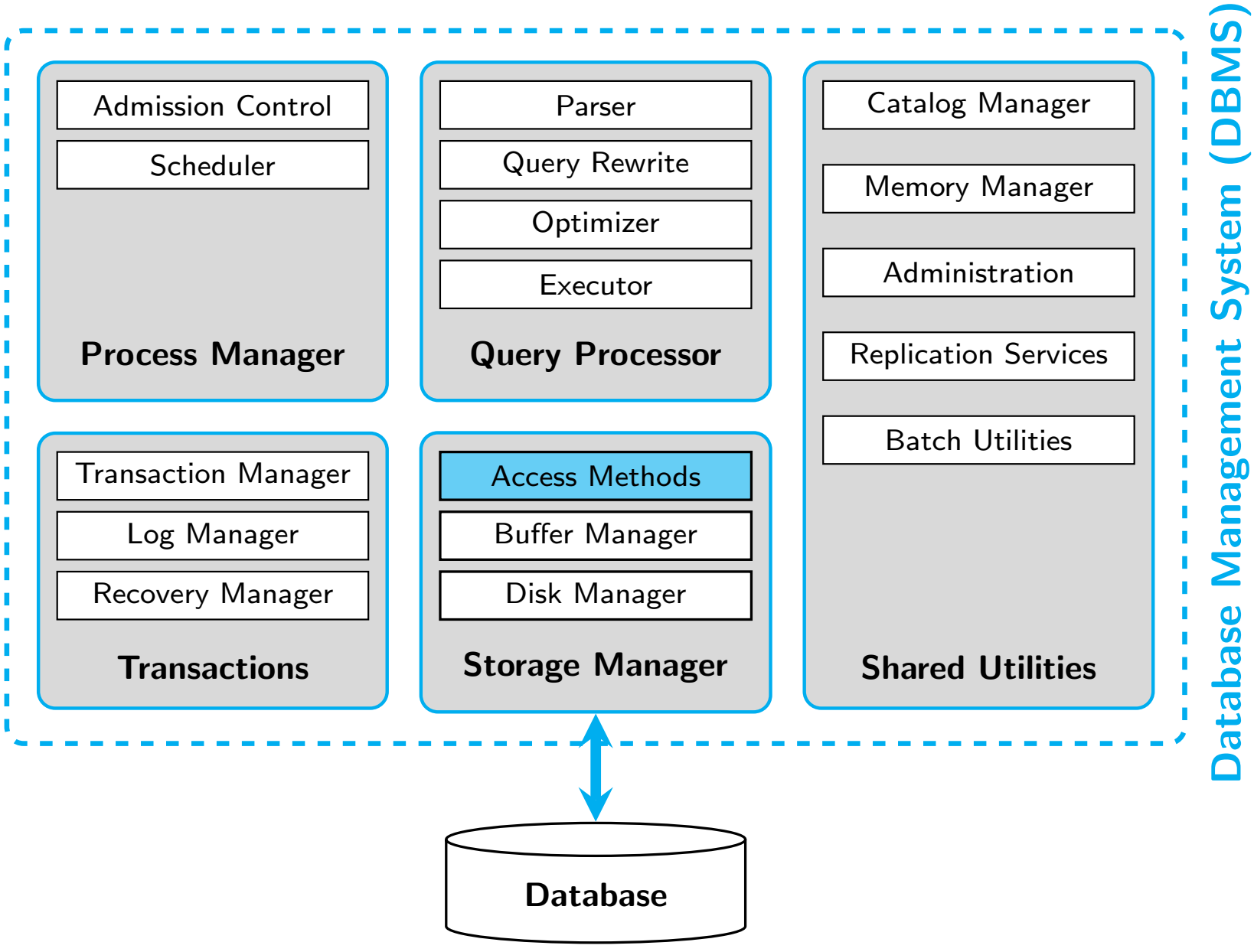
- Mayor conocimiento del patrón de acceso a los datos.
- Política de reemplazo propia.
- Hacer pinned y unpinned de las páginas.
- Forzar el almacenamiento de páginas.
- Acceso “fino” por parte del administrador de transacciones.

Outline

Buffer Manager

Índices

Métodos de acceso



Métodos de acceso

Procesador de consultas
considera las relaciones como "colección de records".

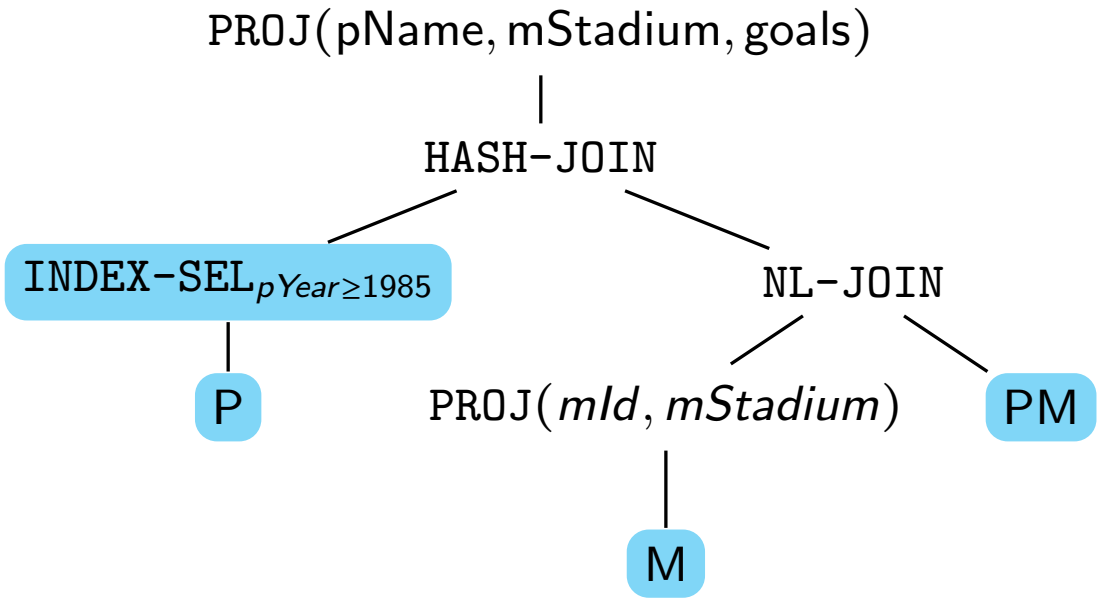
Administrador de almacenamiento provee al procesador de consultas con métodos para acceder estas colecciones de records.

Estos métodos son conocidos como **métodos de acceso**.

Evaluación de consulta y métodos de acceso

Q = `SELECT` pName, mStadium, goals
`FROM` Players `AS` P, Matches `AS` M, Players_Matches `AS` PM
`WHERE` P.pId = PM.pId `AND` PM.mId = M.mId `AND`
P.pYear $\geq$ 1985

Plan físico para evaluar Q:



Interfaz de acceso a relaciones

1. **Create** o **Destroy**
   - Crear o destruir el acceso.
2. **Insert**(record)
   - Insertar un nuevo record a la relación.
3. **Delete**(RID)
   - Eliminar un record dado su Record ID (RID).
4. **Get**(searchkey)
   - Buscar un record dado una "llave de busquedad".
   - El criterio de busqueda puede ser diverso (por valor, por rango, etc).
   - En general, **Get** puede estar "sobrecargado".
5. **Scan**
   - Iterar sobre todos los records.

Ejemplo: Heapfile

1. **Create** o **Destroy**: crea un heap-file de la relación.
2. **Insert**(record): inserta un record al final del archivo.
3. **Delete**(RID): busca la página del record y la elimina.
4. **Get**(searchkey)
   - Busca en todas las páginas los records que satisface el searchkey.
5. **Scan**
   - Iterar sobre todas las páginas y sobre todos los records.

Ejemplo: Sortedfile

1. **Create** o **Destroy**: crea un sortedfile de la relación.
2. **Insert**(record): inserta un record en la **posición** que le corresponde.
3. **Delete**(RID): busca la página del record y la elimina.
4. **Get**(searchkey)
   - **Busqueda binaria** del record que satisface el searchkey.
5. **Scan**.
   - Iterar sobre todos las páginas y sobre todos los records.

¿qué es un índice?

Definición

Método de acceso que **optimiza** el acceso a los datos
para **una consulta o conjunto de consultas** en particular.

Ejemplos

- Índice de un libro.
- Orden alfabético en un diccionario.
- Número de páginas de un libro.
- Secciones de un diario.

Algunos hechos importantes sobre un índice

1. Un índice optimiza un subconjunto de consultas.

2. Un índice optimiza ciertas consultas
   pero puede hacer otras más costosas.

3. Es posible sacrificar espacio por tiempo,
   pero **idealmente** un índice debe ser mantenido en RAM.

¿qué consultas nos gustaría optimizar con índices?

Busqueda por **valor** (*value query*).

```
SELECT    *
FROM      table
WHERE     attribute = 'value'
```

Busqueda por **rango** (*range query*).

```
SELECT    *
FROM      table
WHERE     attribute ≥ 'value'
```

Busqueda por **match** (*pattern matching*).

```
SELECT    *
FROM      table
WHERE     attribute LIKE 'patrón'
```

¿qué otra consulta nos gustaría optimizar?

Evaluación de la eficiencia de índices

Evaluación con respecto a:

- Tipo de acceso.
- Tiempo de acceso.
- Tiempo de inserción.
- Tiempo de eliminación.
- Sobrecarga de espacio.

¿qué parametro es el más importante?

Componentes de un índice

Definición

Search key = parámetros de busqueda.

Index entry = valor o puntero
de la estructura interna de un índice

Data entry = record mismo, o
dirección donde se almacena un record.

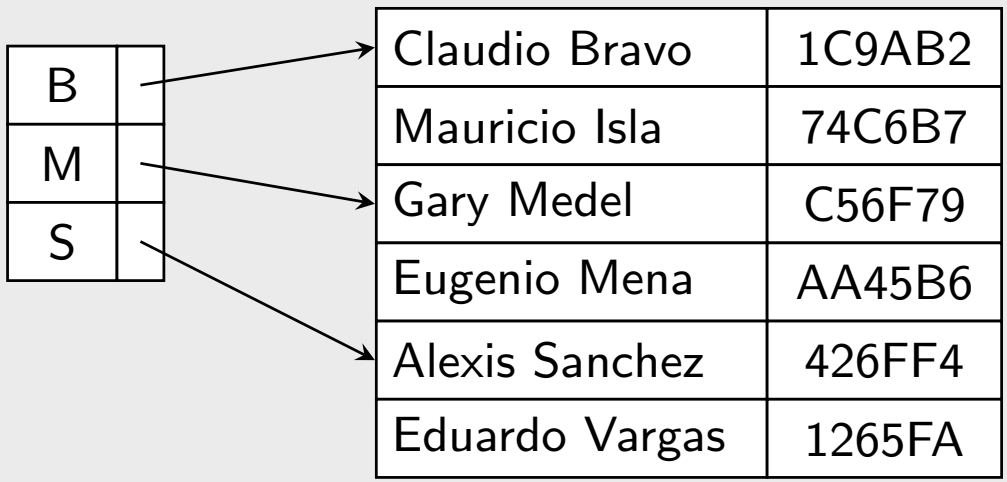
Para un search key k, un **data entry** puede ser:

1. Un record. (que satisface el search key k)
2. (k, RID).
3. $(k,$ lista de $RID)$.

Componentes de un índice

Ejemplos

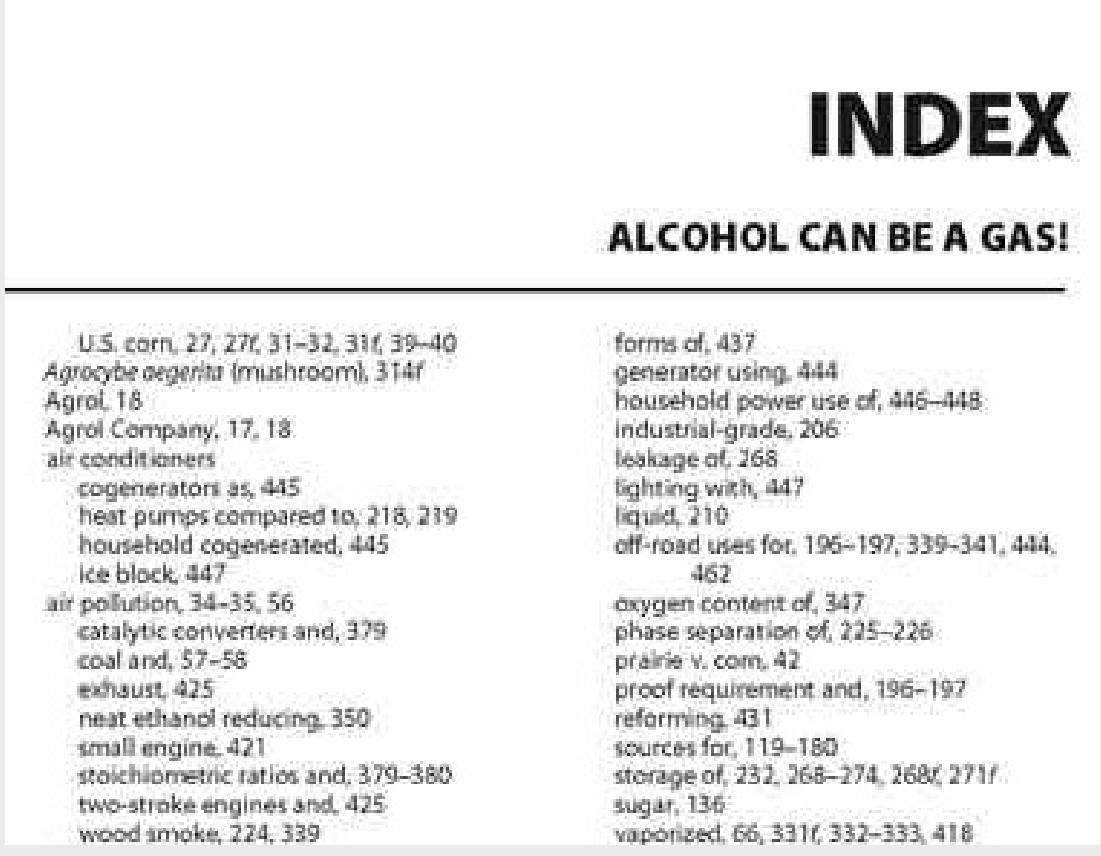
Índice sobre la relación *Players(ID, name, number)*:



¿cuáles son los **search keys**, **index** y **data entries** de este índice?

Componentes de un índice

Ejemplos



INDEX

ALCOHOL CAN BE A GAS!

U.S. corn, 27, 27*f*, 31–32, 31*f*, 39–40
Agrocybe aegerita (mushroom), 314*f*
Agrol, 18
Agrol Company, 17, 18
air conditioners
  cogenerators as, 445
  heat pumps compared to, 218, 219
  household cogenerated, 445
  ice block, 447
air pollution, 34–35, 56
  catalytic converters and, 379
  coal and, 57–58
  exhaust, 425
  neat ethanol reducing, 350
  small engine, 421
  stoichiometric ratios and, 379–380
  two-stroke engines and, 425
  wood smoke, 224, 339
forms of, 437
generator using, 444
household power use of, 446–448
industrial-grade, 206
leakage of, 268
lighting with, 447
liquid, 210
off-road uses for, 196–197, 339–341, 444, 462
oxygen content of, 347
phase separation of, 225–226
prairie v. corn, 42
proof requirement and, 196–197
reforming, 431
sources for, 119–180
storage of, 232, 268–274, 268*f*, 271*f*
sugar, 136
vaporized, 66, 331*f*, 332–333, 418

¿cuáles son los **search keys**, **index** y **data entries** de este índice?

Componentes de un índice

Ejemplos

Relación *Players(ID, name, number)*, ordenado por ID:

232	Claudio Bravo	1
335	Gary Medel	17
481	Eugenio Mena	2
520	Mauricio Isla	4
555	Eduardo Vargas	11
630	Alexis Sanchez	7

¿cuáles son los **search keys**, **index** y **data entries** de este índice?

Clustered indexes

Definición

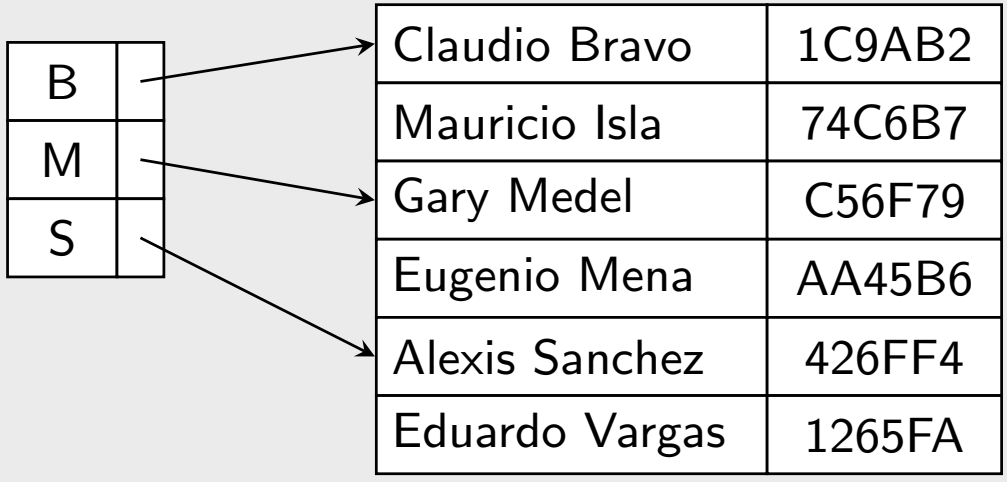
Clustered index = índice para el cual el orden de sus data entries es el mismo orden de los records en disco.

Unclustered index = índice que NO es clustered.

Clustered indexes

Ejemplos

Índice sobre la relación *Players(ID, name, number)*:

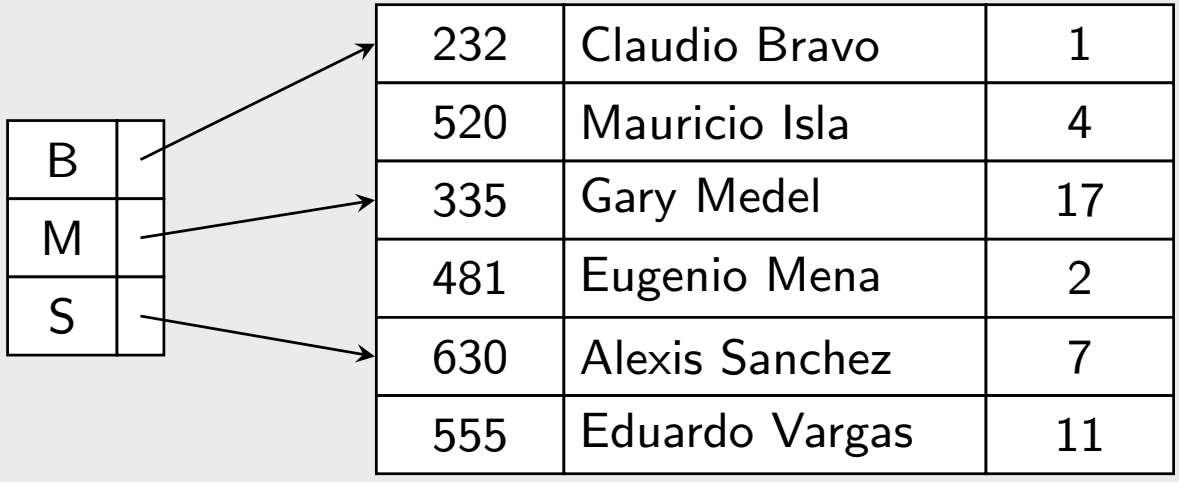


Suponiendo que la relación esta almacenada por orden de ID,
¿es este índice **clustered** o **unclustered**?

Clustered indexes

Ejemplos

Índice sobre la relación *Players(ID, name, number)*:



¿es este índice **clustered** o **unclustered**?
¿qué tipo de **data entry** tiene este índice?

Clustered indexes

- En general, por cada relación
  es posible mantener un solo clustered index. (¿por que?)
- Unclustered index tiene data entries del tipo 2 o 3.
- Unclustered index son ineficientes cuando el output es numeroso.
  - range queries.
- Usualmente, clustered y unclustered indexes son conocidos como:

  clustered index = índice **primario**.

  unclustered index = índice **secundario**.

Índices densos o dispersos

Definición

Los índices pueden ser:

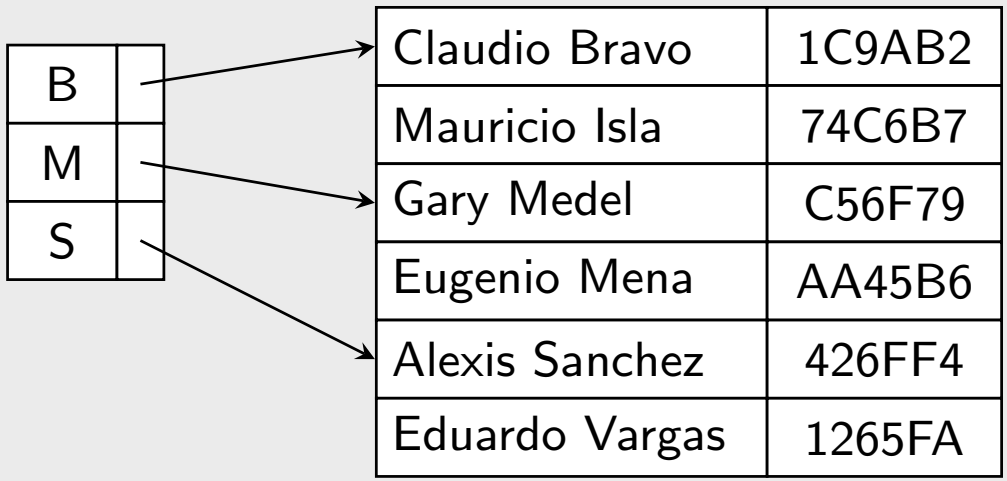
Denso (dense) = un index entry por cada record de la relación.

Disperso (sparse) = no todos los records están mencionadas en los index entries.

Índices densos o dispersos

Ejemplos

Índice sobre la relación *Players(ID, name, number)*:

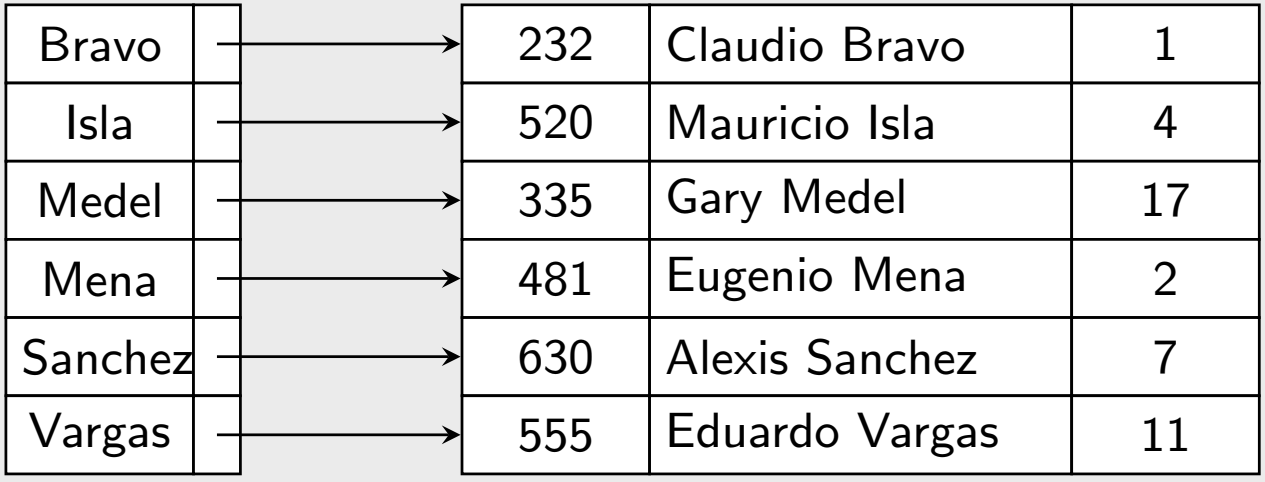


¿es este índice **denso** o **disperso**?

Índices densos o dispersos

Ejemplos

Índice sobre la relación *Players(ID, name, number)*:



¿que ventaja puede tener este índice?

Resumen de la clasificación de indices

Clustered vs unclustered

- Clustered: el orden de sus data entries es el mismo que los records.
- Unclustered: data entries no mantienen el mismo orden de los datos.

Denso vs disperso

- Denso: un index entry por cada record.
- Disperso: no todos los records están mencionadas en los index entries.

Dos tipos de índices básicos

1. Índices basados en **árboles**:
   uso del **orden** de los valores para organizar los records.
   - ISAM.
   - B+ trees.

2. Índices basados en **hashing**:
   uso de una distribución uniforme de los valores sobre distintos grupos.
   - Extendable Hashing.