

# Storing Assignment Project Exam Help and Files

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# 11.1 Memory Hierarchy

- *Primary Storage*: main memory.

fast access, expensive

Assignment Project Exam Help

- *Secondary storage*

<https://eduassistpro.github.io/>

slower access, less expensive.

Add WeChat edu\_assist\_pro

- *Tertiary storage*: tapes, cd, etc.

slowest access, cheapest.

# 11.2 Disks

Characteristics of disks:

- collection of platters
- each platter = set of sectors
- each track = sequence of sectors
- transfer unit: 1 block (e.g. 512B, 1KB)
- access time depends on proximity of heads to required block
- access via block address (p, t, s)

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# 11.2 Disks

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

- Data must be in memory for the DBMS to operate on it.
- If a single record in a block is needed, the entire block is transferred.

# 11.2 Disks

Access time includes:

- seek time (find the right track, e.g. 10*msec*)
- rotational delay (e.g. 5*msec*)
- transfer time (read/write block, e.g. 1*msec*)

➔ Random access is dominated by seek time and rotational delay

# 11.3 Disk Space Management

Disk space is managed by the disk space manager.

1. *Improving Disk Access:*

Assignment Project Exam Help

Use knowledge of data

<https://eduassistpro.github.io/>

E.g. two records often

Add WeChat edu\_assist\_pro

⇒ put them in the same block (cluster)

E.g. records scanned sequentially

⇒ place them in consecutive sectors on same track

# 11.3 Disk Space Management

## 2. *Keeping Track of Free Blocks*

- Maintain a list of free blocks.

Assignment Project Exam Help

- Use bitmap.

<https://eduassistpro.github.io/>

## 3. *Using OS File System to Manage Disk*

Add WeChat edu\_assist\_pro

- extend OS facilities, but
- not rely on the OS file system.

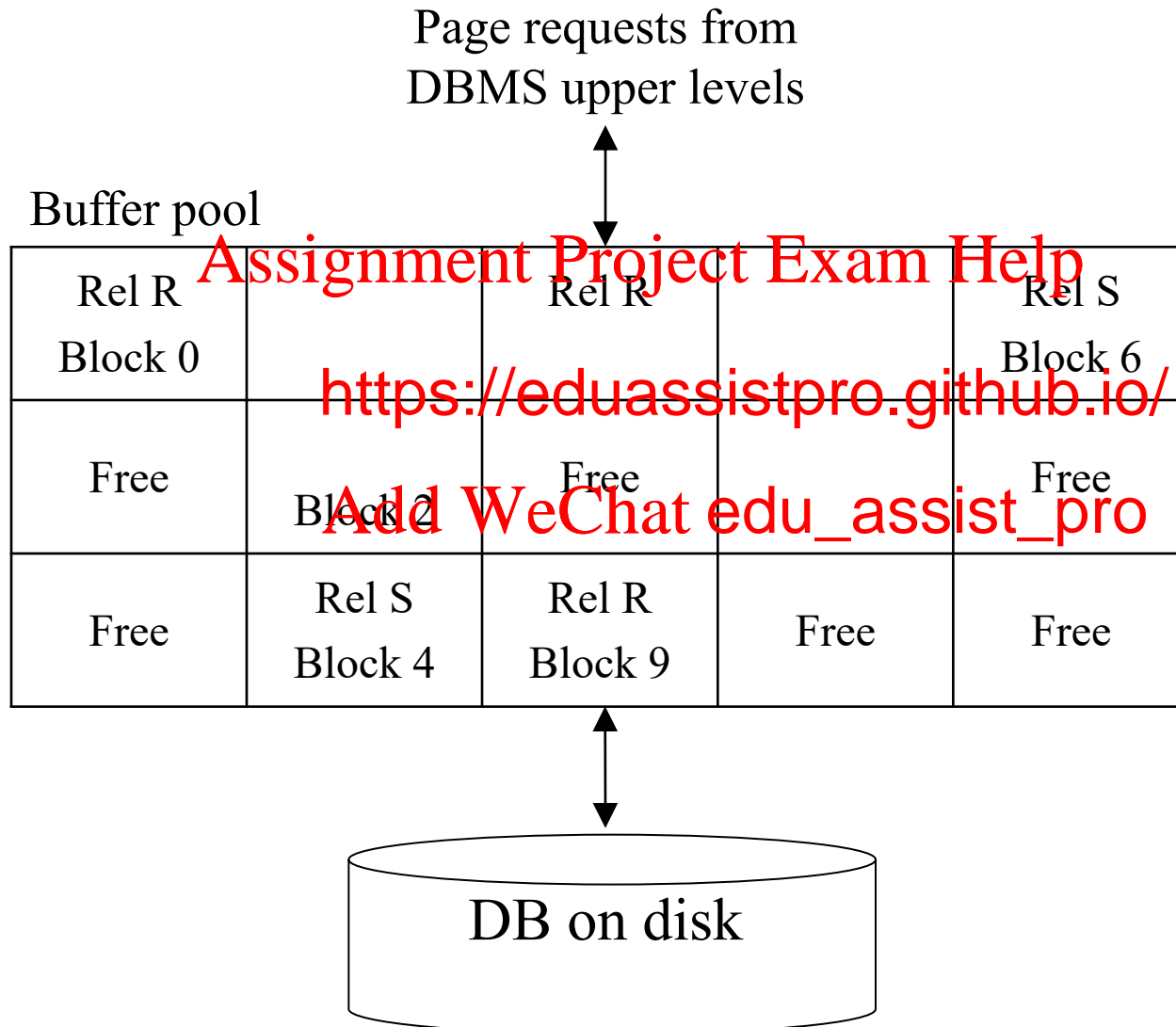
(portability and scalability)

# 11.4 Buffer Management

- *Buffer Manager*
- Manages traffic between disk and memory by maintaining a <https://eduassistpro.github.io/>
- Buffer pool = collection of (frames) which can be filled with copies of disk block data.



# 11.4.1 Buffer Pool



# 11.4.1 Buffer Pool

- The *request\_block* operation replaces *read\_block* in all file access algorithms.
- If block is already in buffer pool:
  - no need to read it again
  - use the copy there (u
- If block is *not* already in buffer pool:
  - need to read from hard disk into a free frame
  - if no free frames, need to remove block using a *buffer replacement policy*.
- The *release\_block* function indicates that block is no longer in use  $\Rightarrow$  good candidate for removal.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# 11.4.1 Buffer Pool

For each frame, we need to know:

- whether it is currently in use
- whether it has *dirty bit*
- how many transactions are using it (*pin count*)
- (maybe) time-stamp for most recent access

# 11.4.1 Buffer Pool

## The *request\_block* Operation

Method:

1. Check buffer pool to see if it already contains requested block.

If not, the block is brought in as follows:

- (a) Choose a frame for replacement
  - (b) If frame chosen is dirty, write block to disk
  - (c) Read requested page into now-vacant buffer frame
- = False and pinCount = 0)*
2. *Pin* the frame containing requested block.  
(This simply means updating the pin count.)
  3. Return address of frame containing requested block.

# 11.4.1 Buffer Pool

## The *release\_block* Operation

Method:

1. Decrement pin count for specified page

No real effect until replac

## The *write\_block* Operati

Method:

1. Updates contents of page in pool
2. Set dirty bit on

Note: Doesn't actually write to disk.

The *force\_block* operation “commits” by writing to disk.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# 11.4.2 Buffer Replacement Policies

Several schemes are commonly in use:

- Least Recently Used (LRU)
  - release the frame that has not been used for the longest period.
  - intuitively appealing idea but can perform badly
- First in First Out (FIFO)
  - need to maintain a queue of frames
  - enter tail of queue when read in
- Most Recently Used (MRU): release the frame used most recently
- Random

No is guaranteed better than the other.

For DBMS, we may predict accesses better.

Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

Example1:

Data pages: P1, P2, P3, P4

Q1: read P1; Q2: read P2;

Q3: read P3; Q4: read P1;

Q5: read P2; Q6: read P4;

Buffer:

|          |          |    |
|----------|----------|----|
| P1<br>Q4 | P2<br>Q5 | P3 |
|----------|----------|----|

Regarding Q6,

- LRU: Replace P3
- MRU: Replace P2
- FIFO: Replace P1
- Random: randomly choose one buffer to replace

Example 2:

Data pages: P1, P2, ..., P11

10 buffer pages as in Example 1

Q1: read P1, P2,..., P11;

Q2, read P1, P2,..., P11;

Q3: Read P1, P2,...,P11

LRU/FIFO: I/O P1, P2, ..., P11 for each query.

Assignment Project Exam Help

<https://eduassistpro.github.io/> is the best

Add WeChat edu\_assist\_pro

# 11.5 Record Formats

Records are stored within fixed-length blocks.

- *Fixed-length*: each field has a fixed length as well as the number of fields.
  - Easy for intra-block space management.
  - Possible waste of
- *Variable-length*: so
  - complicates intra-block space management
  - does not waste (as much) space.

Record format info:

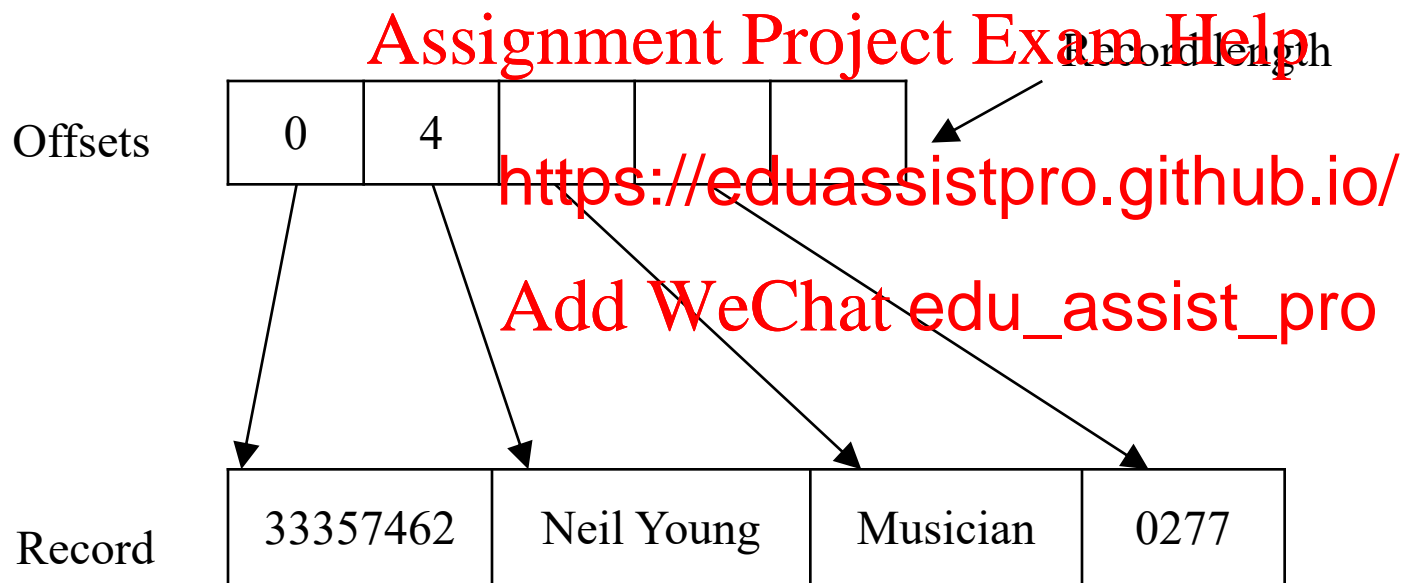
- best stored in data dictionary
- with dictionary memory-resident



# 11.5.1 Fixed-Length

Encoding scheme for fixed-length records:

- length + offsets stored in header



# 11.5.2 Variable-Length

Encoding schemes for variable-length records:

- Prefix each field by length

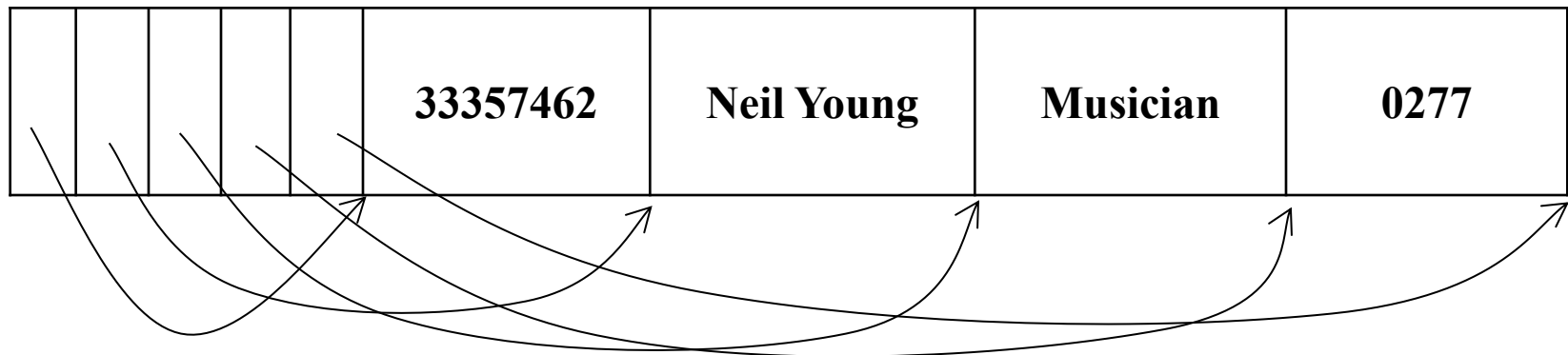
**Assignment Project Exam Help**  

|   |      |    |            |   |          |   |      |
|---|------|----|------------|---|----------|---|------|
| 4 | xxxx | 10 | Neil Young | 8 | Musician | 4 | xxxx |
|---|------|----|------------|---|----------|---|------|

- Terminate fields <https://eduassistpro.github.io/>

33357462/Neil Young/Musician/0  
**Add WeChat edu\_assist\_pro**

- Array of offsets



# 11.6 Block (Page) Formats

A block is a collection of *slots*.

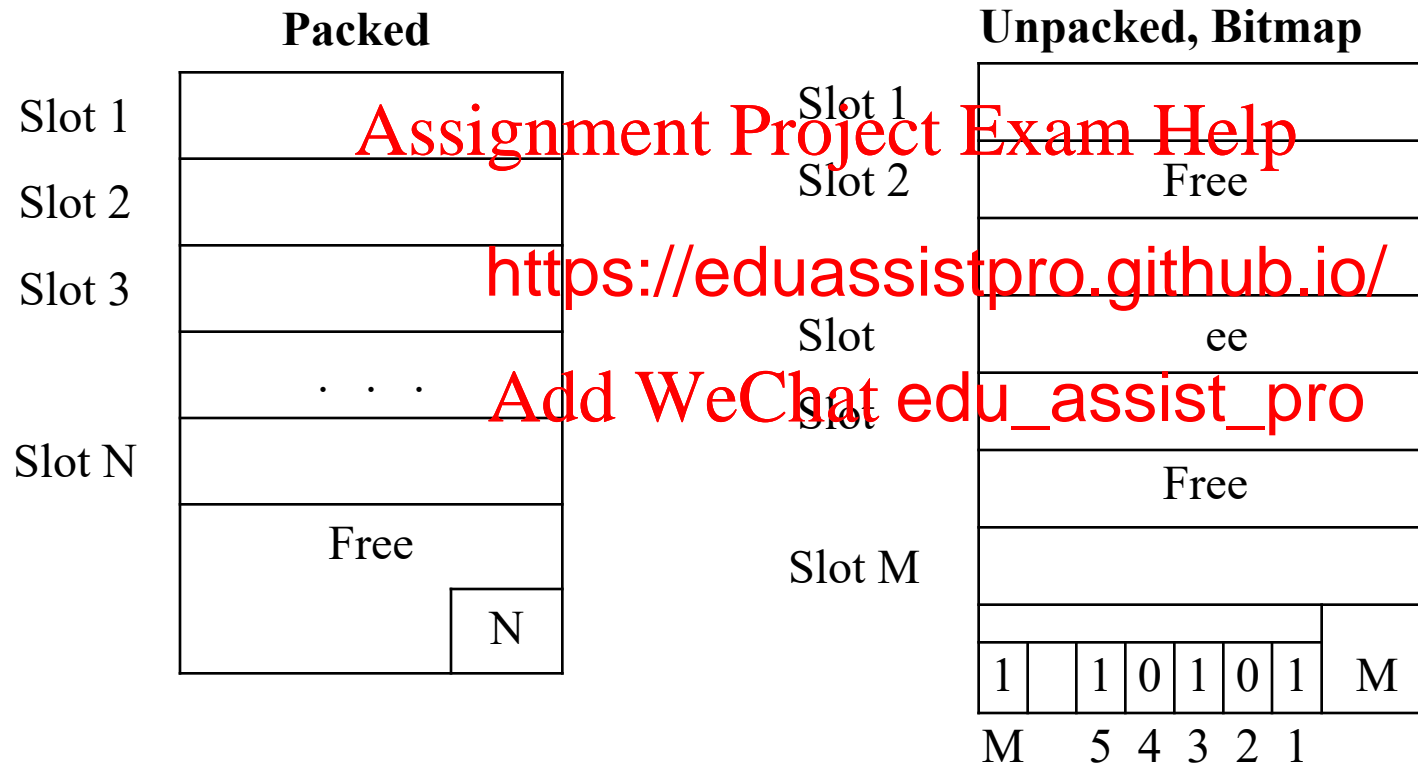
Each slot contains a record.

A record is identified by its slot number  $>$ .

<https://eduassistpro.github.io/>  
Add WeChat edu\_assist\_pro

# 11.6.1 Fixed Length Records

For fixed-length records, use record slots:



Insertion: occupy first free slot; packed more efficient.

Deletion: (a) need to compact, (b) mark with 0; unpacked more efficient.

# 11.6.2 Variable-Length Records

For variable-length records, use slot *directory*.

Assignment Project Exam Help

Possibilities for handling

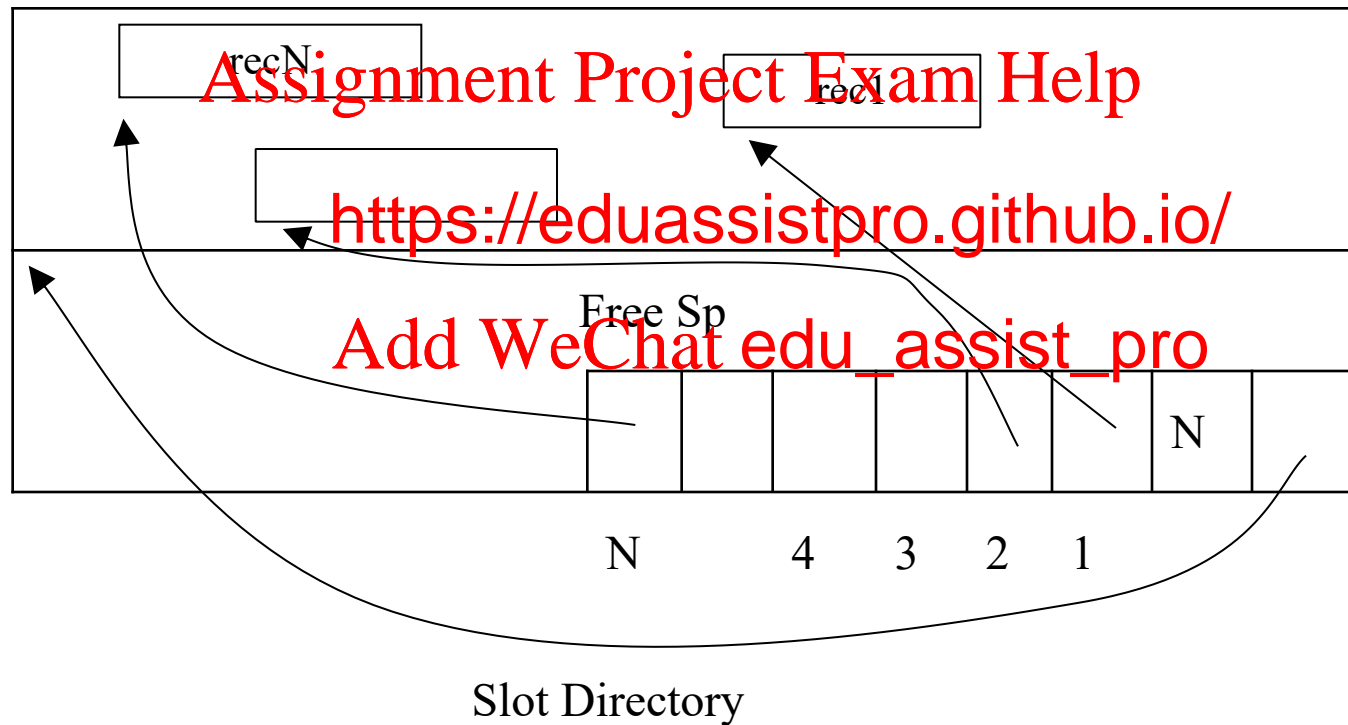
- compacted (one record per slot)
- fragmented (distributed free space)

In practice, probably use a combination:

- normally fragmented (cheap to maintain)
- compact when needed (e.g. record won't fit)

# 11.6.2 Variable-Length Records

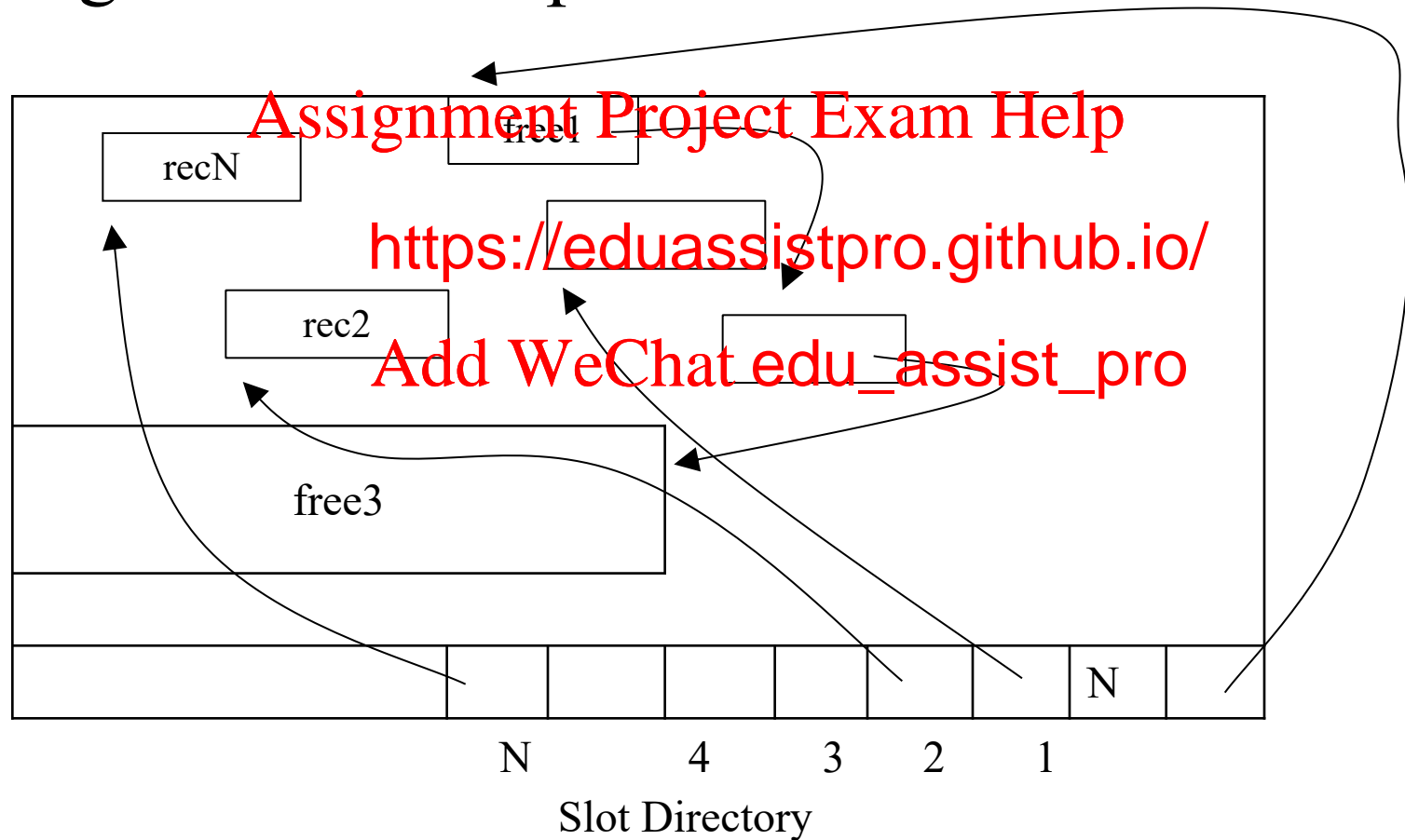
- Compacted free space:



- Note: “pointers” are implemented as offsets within block; allows block to be loaded anywhere in memory.

# 11.6.2 Variable-Length Records

- Fragmented free space:



# 11.6.2 Variable-Length Records

## Overflows

Some file structures (e.g. hashing) allocate records to specific blocks.

What happens if specified block is already full?

Need a place to store “ex

<https://eduassistpro.github.io/>

Introduce notion of *overflow blocks*. Add WeChat edu\_assist\_pro

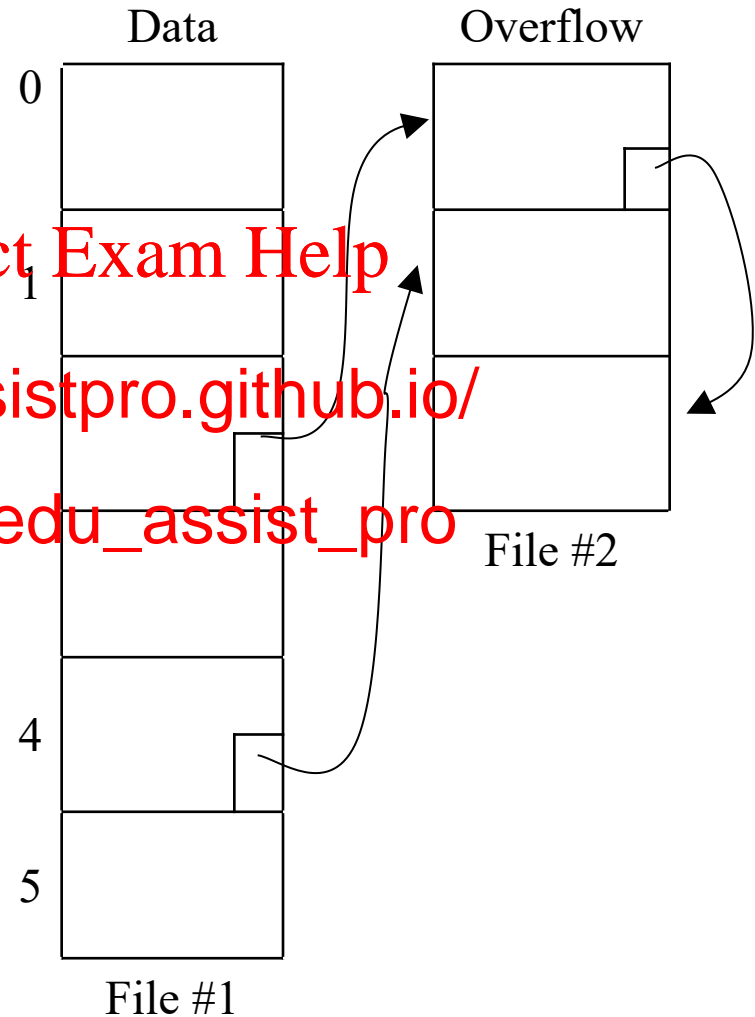
- located outside main file (don’t destroy block sequence of main file)
- connected to original block
- may have “chain” of overflow blocks

New blocks are always appended to file.



# 11.6.2 Variable-Length Records

- Overflow blocks in a separate file:
- Note: “pointers” are implemented as file offsets.



Assignment Project Exam Help

<https://eduassistpro.github.io/>

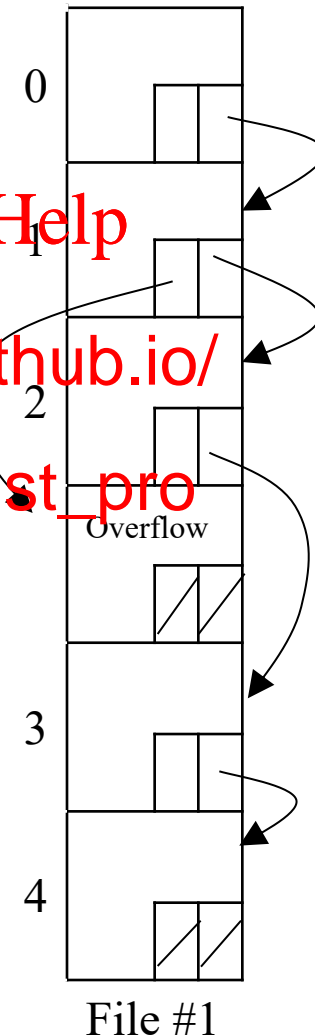
Add WeChat edu\_assist\_pro

# 11.6.2 Variable-Length Records

- Overflow blocks in a single file:

- Not suitable if accessing blocks via offset (e.g. hashing).

**Data + overflows**



Assignment Project Exam Help

<https://eduassistpro.github.io/>

Add WeChat edu\_assist\_pro

# 11.7 Files

A *file* consists of several data blocks.

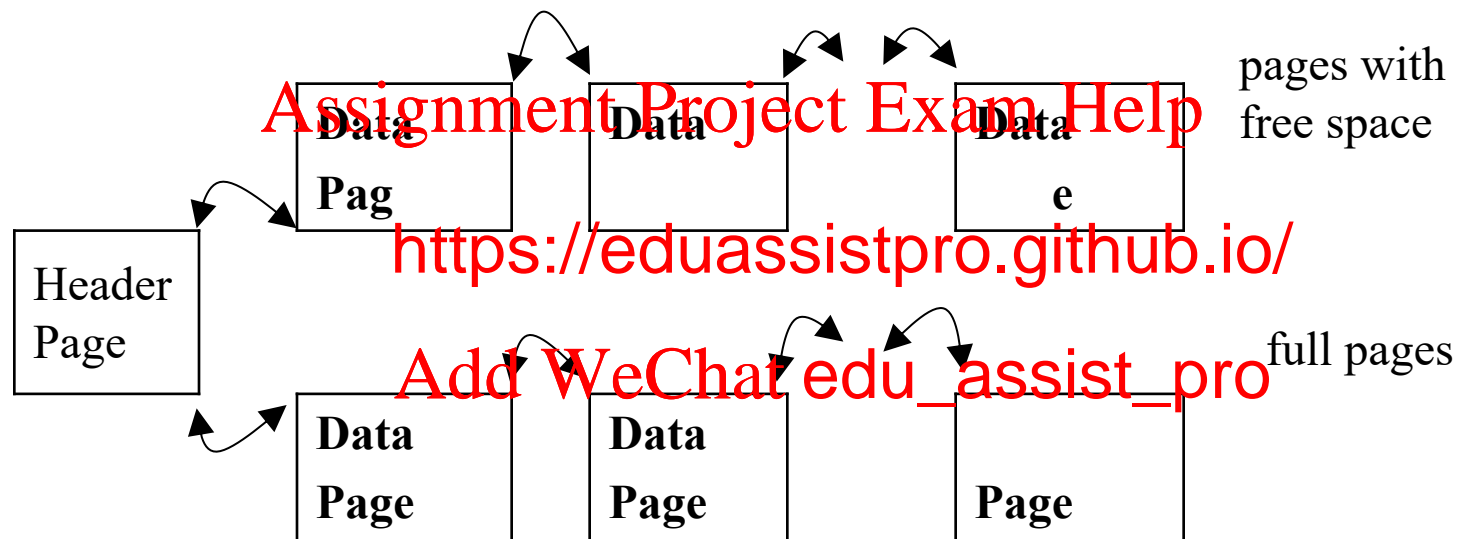
*Heap Files*: unordered pages (blocks).

Two alternatives <https://eduassistpro.github.io/> information:

- Linked list of pages.
- Directory of pages.

# 11.7.1 Linked List of Pages

- Maintain a heap file as a doubly linked list of pages.



Organized by a Linked List

- Disadvantage:** all pages will virtually be on the free list of records if records are of variable length. To insert a record, several pages may be retrieved and examined.

# 11.7.2 Directory of Pages

Maintain a directory of pages.

- Each directory entry identifies a page (or a sequence of pages) in the heap file.
- Each entry also maintains a bit to indicate if the corresponding page has any free space.

