# Buffer Pool Manager

The buffer pool is an in-memory cache of pages read from disk. The DBMS often just mallocs a large memory cache for the buffer pool on start up that is of fixed size. You can often configure the size of this space. This space is usually organized as an array of frames. On a page request, an exact copy is placed into one of these frames from disk. The Buffer Pool Manager manages the placement and replacement of pages in the buffer.

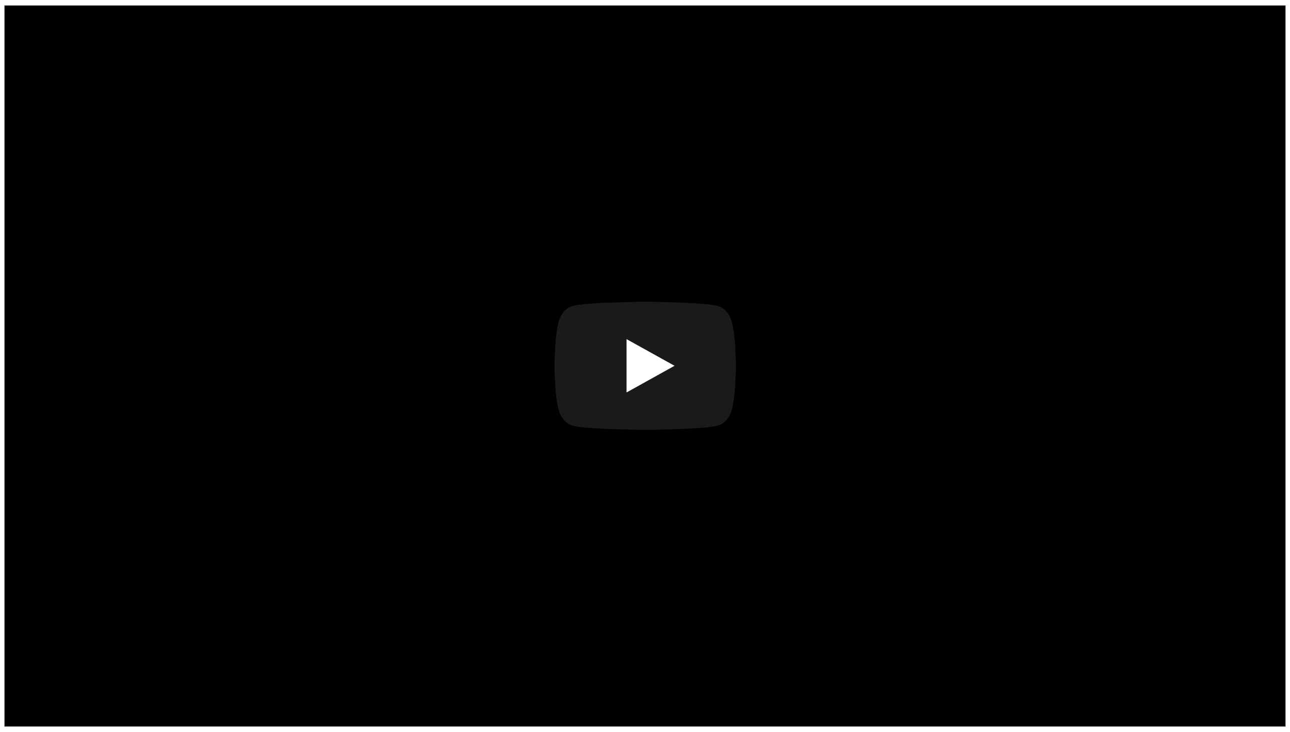
**Reading Material:** Chapter 9.4 of Ramakrishnan and Gerhke

As you go through the material, answer all the relevant assessment questions. The deadline for submission of this assessment is Tuesday, March 24 @ 1:00 pm. You can update your answers as often as you want. It is not a timed assessment.

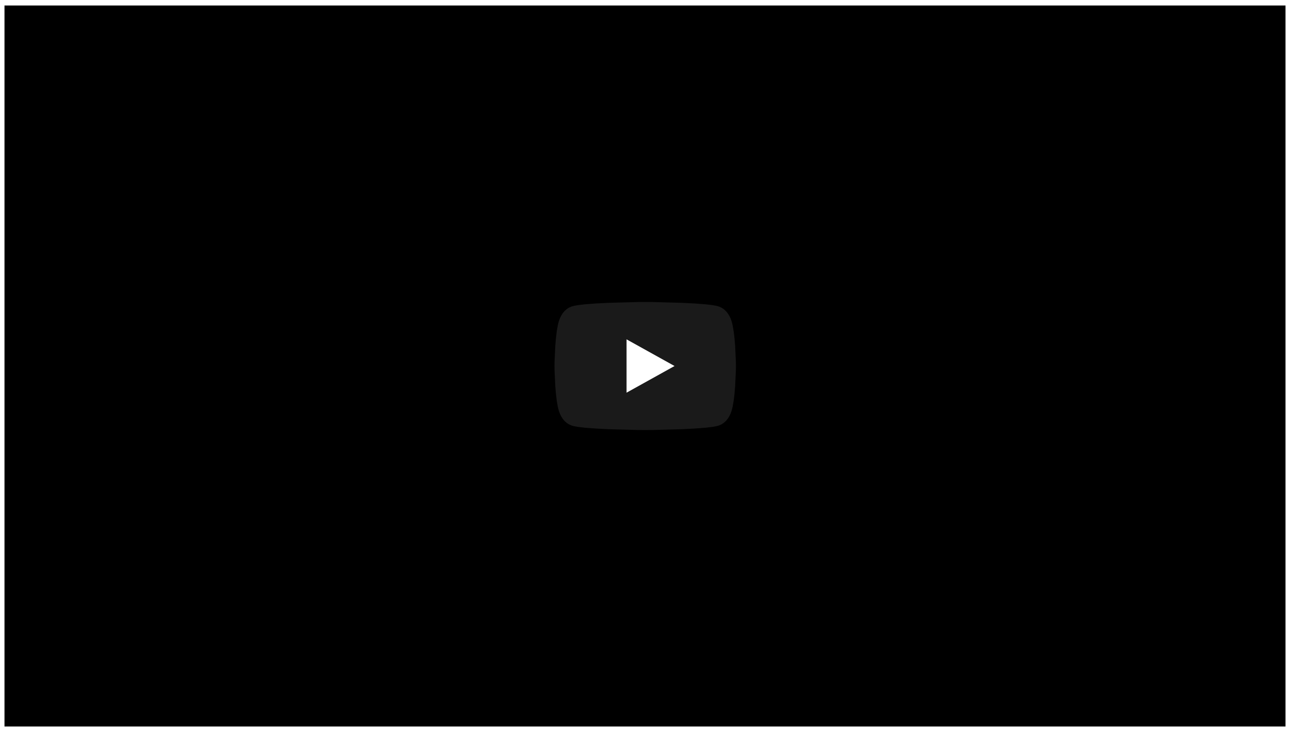
**Assessment:** Complete the buffer pool manager assessment in GradeScope.

- Buffer Pool Manager Assessment

## The Job of a Buffer Manager



## Moving pages in an out of the Buffer Pool

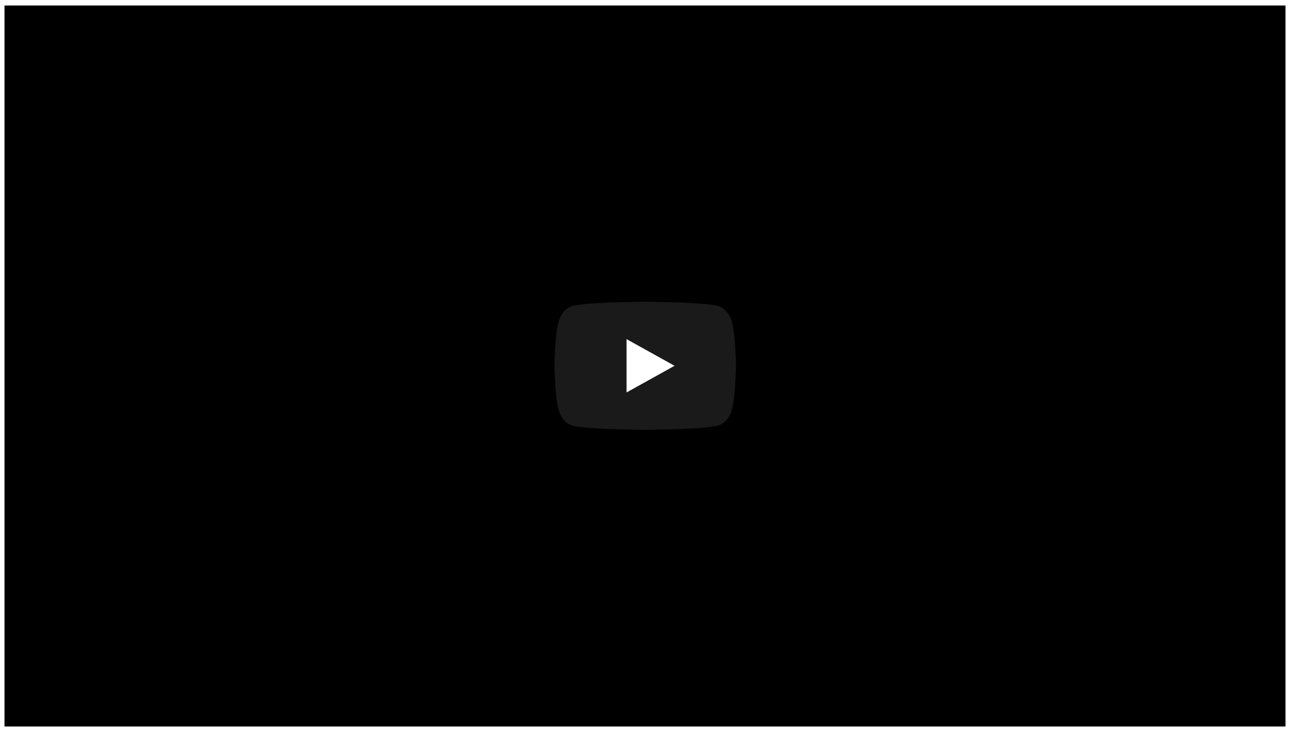


**Did you watch the above videos?**

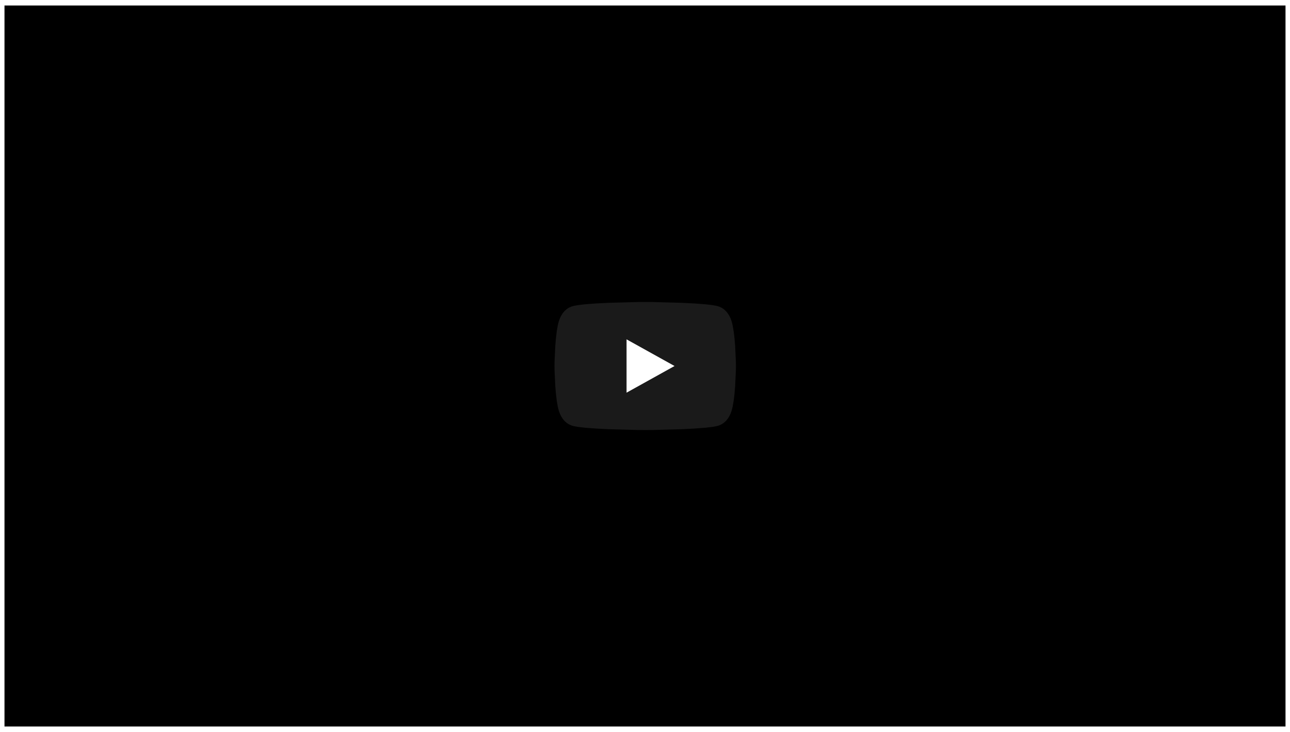
- A. Yes
- B. No

Submit Answer

## Dealing with Page Updates: Dirty Pages



## Buffer Page Meta Data



**Did you watch the videos above?**

- A. Yes
- B. No

Submit Answer

To summarize the meta-data maintained by the buffer pool:

- **Page Table:** In-memory hash table that keeps track of pages that are currently in memory. It maps page ids to frame locations in the buffer pool.
- **Dirty-flag:** When a page is modified by the DBMS threads, this flag is set. A dirty page must be written back to disk.
- **Pin Counter:** This tracks the number of threads that are currently accessing that page (either reading or modifying it). A thread has to increment the counter before they access the page. If a page's count is greater than zero, then we cannot evict that page from memory.

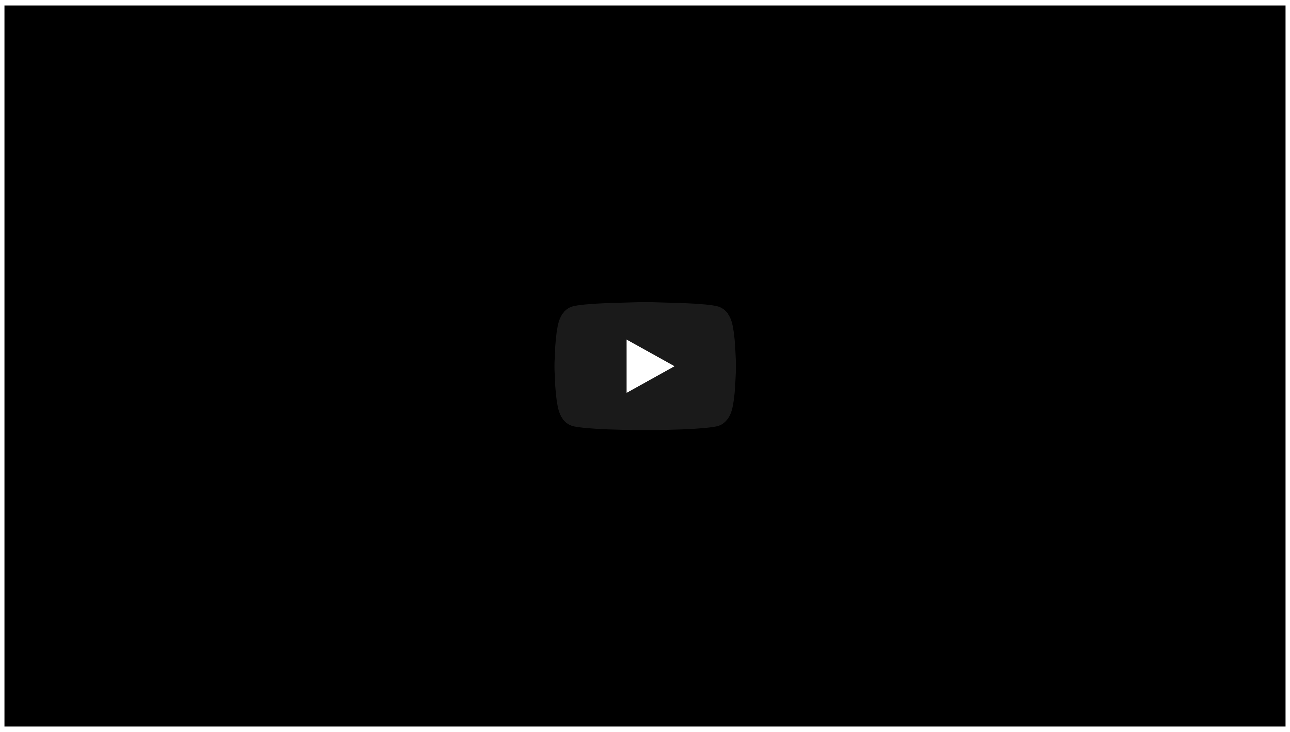
Now you can complete questions 1 and 2 of the Buffer Pool Management Assessment on Gradescope.

## Making Room: Page Replacement Overview

A replacement policy is an algorithm that the DBMS implements that makes a decision on which pages to evict from buffer pool when it needs space.

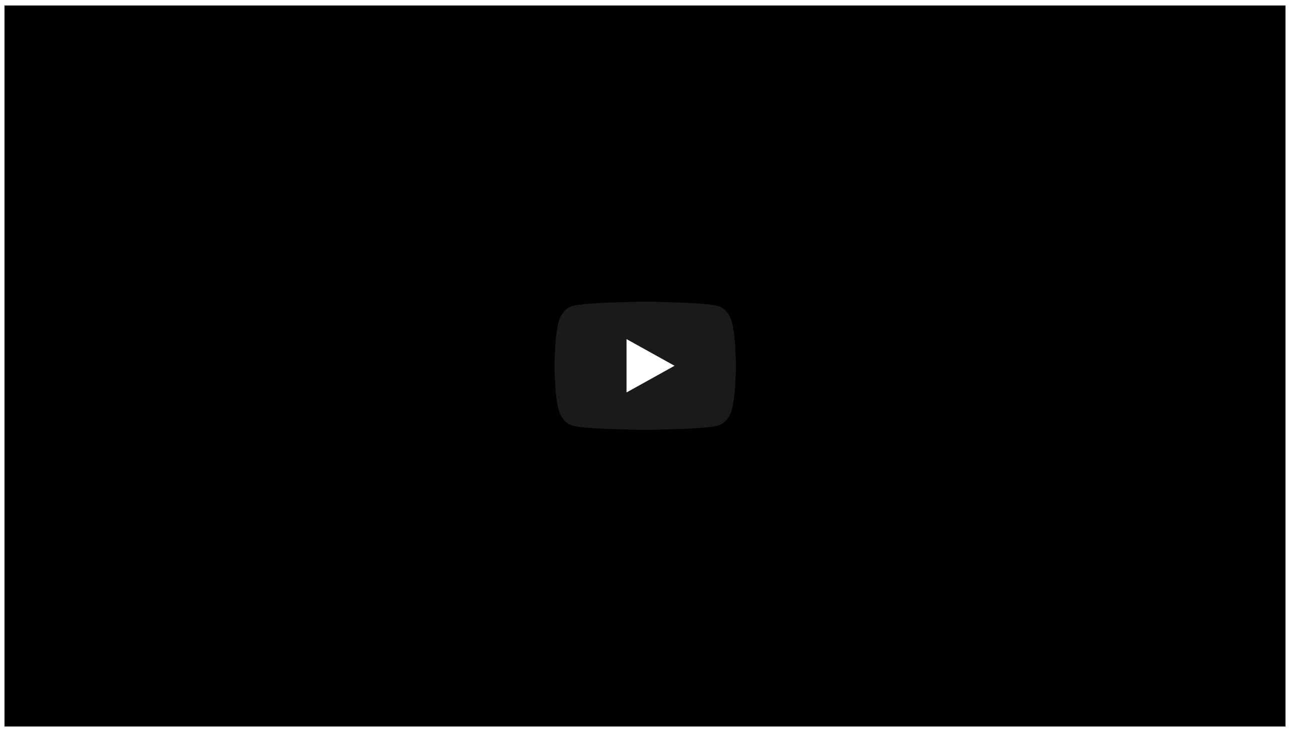
There are several criteria that guide the choice of the policy to use including:

- Correctness - You don't want to evict pinned pages!
- Accuracy - You shouldn't evict pages that you will likely read again soon.
- Speed - Deciding which page to evict should be quick for interactive query performance: you don't want to use an inefficient strategy to decide which pages to replace.
- Meta-data overhead - Some policies like LRU keep a timestamp that tracks when every page was accessed. This overhead is one of the reasons behind its approximation: the clock. You want to strike a balance between how much information you keep and its ability to improve the replacement policy. You are also constrained by how much space you have to store this metadata.

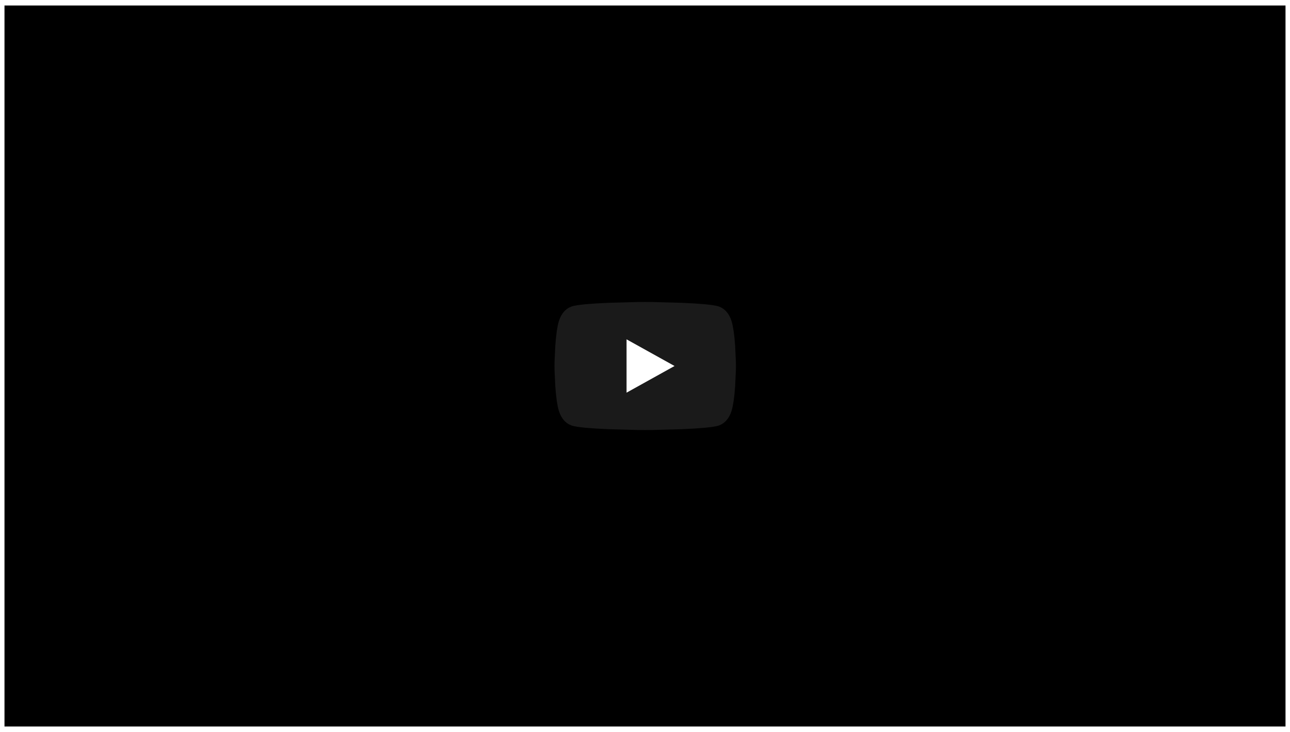


We will now examine two main policies.

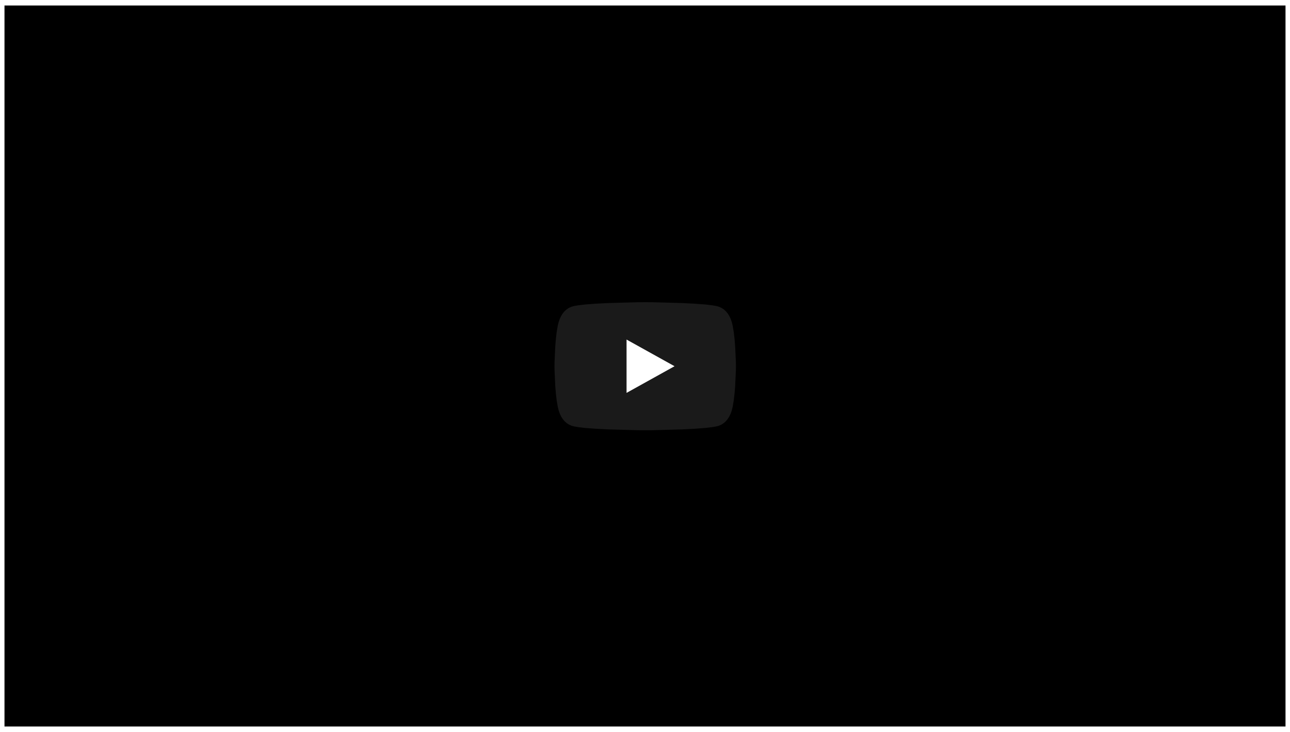
## LRU



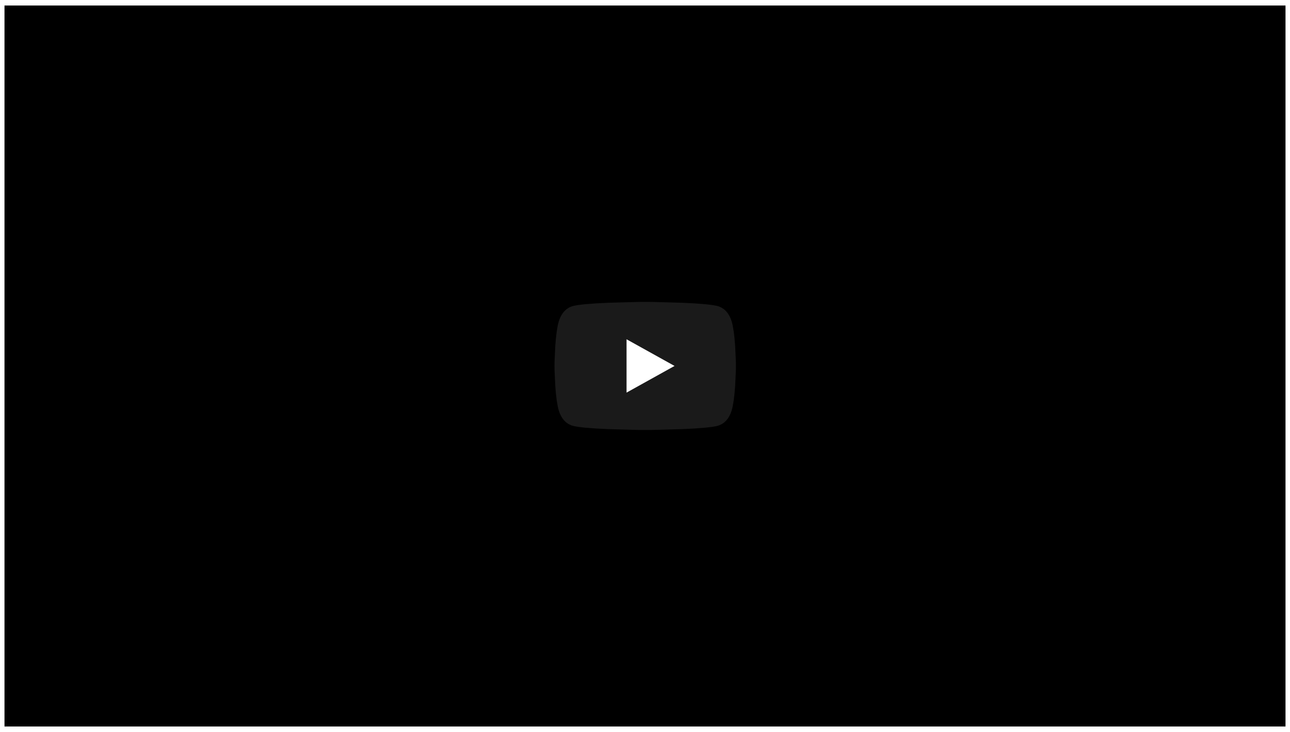
### LRU Approximation: The Clock



### Problematic Access Patterns: LRU & Sequential Flooding



## MRU



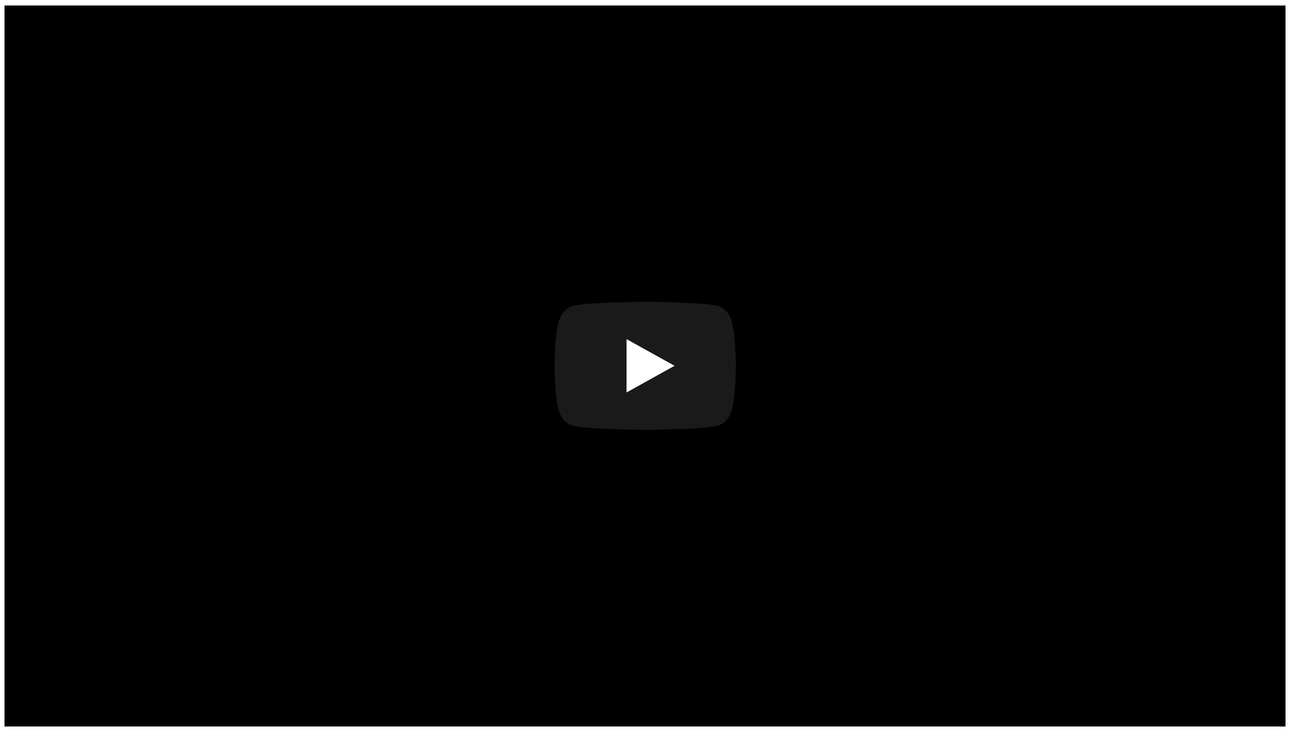
**Did you watch the videos above?**

- A. Yes
- B. No

Submit Answer

Now you can complete questions 3, 4, 5, and 6 of the Buffer Pool Management Assessment on Gradescope.

## Prefetching & Why not just rely on the OS cache?



### Did you watch the video above?

- A. Yes
- B. No

Submit Answer

Now you can complete question 7 of the Buffer Pool Management Assessment on Gradescope.

## For the fun of it

If you would like to understand how caching (buffer pools) work in Postgres and how they interact with OS file caches, read the following blog post: https://madusudanan.com/blog/understanding-postgres-caching-in-depth/

It also describes a number of Postgres commands that can help you debug/improve your query performance if you suspect poor buffering to be the cause of the problem!

**Congrats on completing this unit!** If you have questions/comments regarding this unit please write a Piazza post.

Back Next