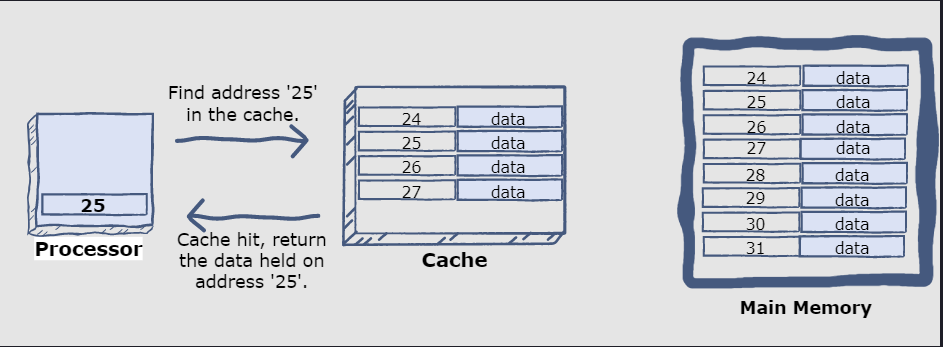
Lecture 6 Notes:

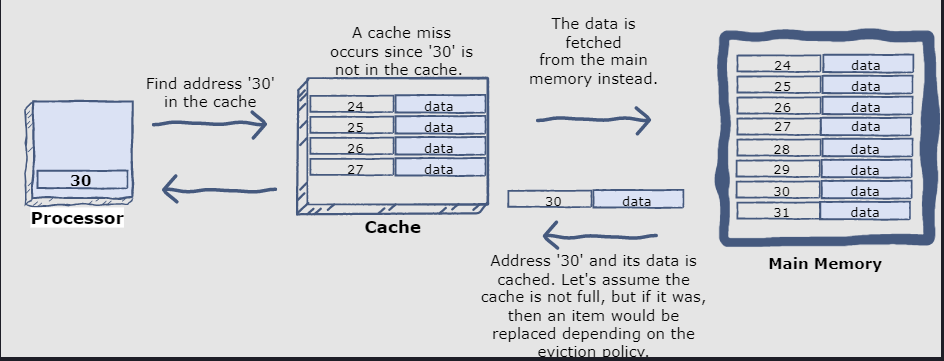
**What is Caching?**  
  
Caches are usually very small in size so that they are cost-effective and efficient. They are frequently used by cache clients such as web-browsers, the CPU, operating systems, and DNS (to name a few). Accessing data from a cache is much faster than accessing it from the main memory or any other type of storage.  
  
**How it works?**  
  
Let’s assume that a cache client wants to access some data. First, the client will check if the data is stored in the cache. If the requested data is found in the cache, it will immediately be returned to the client. This is known as a cache hit.

However, if the data is not stored in the cache, a cache miss occurs. In such insta​nces, the client then fetches the data from the main memory and stores it in the cache. The mechanism for storing ​data in the cache depends on the caching algorithm and policies used.

The following illustration demonstrates how a cache hit in a system cache works:



The illustration below shows how a cache miss works:

  
  
  
 **Cache Write Policies:**  
  
While caching is fantastic, it requires some maintenance to keep the cache coherent with the source of truth (e.g., database). If the data is modified in the database, it should be invalidated in the cache; if not, this can cause inconsistent application behavior.

Solving this problem is known as cache invalidation; there are three main schemes that are used:  
  
**Write-through cache:**   
Under this scheme, data is written into the cache and the corresponding database simultaneously. The cached data allows for fast retrieval and, since the same data gets written in the permanent storage, we will have complete data consistency between the cache and the storage. Also, this scheme ensures that nothing will get lost in case of a crash, power failure, or other system disruptions.Although, write-through minimizes the risk of data loss, since every write operation must be done twice before returning success to the client, this scheme has the disadvantage of higher latency for write operations.

**Write-around cache:**   
This technique is similar to write-through cache, but data is written directly to permanent storage, bypassing the cache. This can reduce the cache being flooded with write operations that will not subsequently be re-read, but has the disadvantage that a read request for recently written data will create a “cache miss” and must be read from slower back-end storage and experience higher latency.

**Write-back cache:** Under this scheme, data is written to cache alone, and completion is immediately confirmed to the client. The write to the permanent storage is done after specified intervals or under certain conditions. This results in low-latency and high-throughput for write-intensive applications; however, this speed comes with the risk of data loss in case of a crash or other adverse event because the only copy of the written data is in the cache.

**Cache eviction policies:**  
  
Following are some of the most common cache eviction policies:

1. First In First Out (FIFO): The cache evicts the first block accessed first without any regard to how often or how many times it was accessed before.
2. Last In First Out (LIFO): The cache evicts the block accessed most recently first without any regard to how often or how many times it was accessed before.
3. Least Recently Used (LRU): Discards the least recently used items first.
4. Most Recently Used (MRU): Discards, in contrast to LRU, the most recently used items first.
5. Least Frequently Used (LFU): Counts how often an item is needed. Those that are used least often are discarded first.
6. Random Replacement (RR): Randomly selects a candidate item and discards it to make space when necessary.

**Content Delivery (or Distribution) Network (CDN)**

CDNs are a kind of cache that comes into play for sites serving large amounts of static media. In a typical CDN setup, a request will first ask the CDN for a piece of static media; the CDN will serve that content if it has it locally available. If it isn’t available, the CDN will query the back-end servers for the file, cache it locally, and serve it to the requesting user.