DJ Oneil      Esten Rye

**Question 1:**

**Question:**

Stonebraker identifies the following types of disk block access for a DBMS:

1) sequential access to blocks which will not be rereferenced
2) sequential access to blocks which will be cyclically rereferenced
3) random access to blocks which will not be rereferenced again
4) random access to blocks for which there is a nonzero probability of rereference

For each disk block access type, state whether or not LRU replacement is appropriate and why (assume all the data cannot simultaneously all fit into memory). If LRU should not be used, suggest another method.

**Answer:**

1) LRU is not appropriate. Assuming that all the referenced data cannot fit into memory, there will come a point where data blocks need to be continually replaced. Because LRU is being used and the data will never be rereferenced, the pages will be replaced in FIFO order. In this case the pages should be tossed immediately. Because the database has information on what pages come next in the data set, these pages should be prefetched as pages are tossed.

2) LRU is not appropriate. If LRU is used on n data blocks that cannot all be fit into memory, then the cyclical accesses will be worse case as all pages will have to be loaded into memory from secondary storage every cycle. The reference pattern would be 1, 2, ..., n, 1, 2, ..., n, ...

3) LRU is not appropriate. This is for the same reason as in (1). When data will not be referenced, it should be tossed immediately and replaced with a page that will be used (or at the very least, tossed immediately so when a new page is needed later, it doesn't have to replace a page first).

4) LRU is appropriate. Because the data blocks will be rereferenced and memory is accessed randomly, it is likely that blocks that are frequently accessed will remain in memory.
**Question 2:**

**Question:**
Stonebraker in his paper “Operating System Support for Database Management” that a server model becomes viable if the operating system provides a message utility that allows for n processes to originate messages to a single destination process; however, this requires the server to implement its own scheduling and multi-tasking, duplicating many operating system facilities. Please list and briefly describe the 3 approaches Stonebraker suggests in his paper to avoid this duplication. Describe the advantages or disadvantages of each implementation.

**Answer:**

1. **First Come First Serve Server (with no internal parallelism)**
   a. The server would read a work request from the message system and executes it to completion before starting another request.
   b. **Advantages/Disadvantages**
      i. No Advantages
      ii. Shorter requests must wait for longer requests to finish before they are allowed to start.
      iii. At any given time only one disk is active, increasing the average response time.

2. **Server Pool Structure**
   a. User processes send work requests to a pool of common servers that share a common lock table.
   b. **Advantages/Disadvantages**
      i. Achieves internal parallelism while avoiding multitasking
      ii. Differs only slightly from the process-per-user model. If a server process is de-scheduled while holding a lock, other processes requiring that lock will back up behind that process until the lock is freed.

3. **Disk Server Structure**
   a. User processes send work requests to a pool of common servers that in turn would send low-level requests to a group of disk processes that would actually perform the IO and handle the locking. Disk processes would serve requests in first come first serve order.
   b. **Advantages/Disadvantages**
      i. If a disk process is de-scheduled, it only blocks those server processes that need to access it. Other server processes which need to access other disk processes will not be affected.
      ii. Achieves internal parallelism while avoiding multitasking
      iii. All we have really done is trade a task switch per I/O for a message per I/O which could end up being more expensive because of the high number of instructions required for a message.