The Bounded-Buffer Problem

- If the buffer is full, the producer must wait until the consumer deletes an item.
  - Producer needs an empty space
  - # of empty slot is represented by a semaphore *empty*
- If the buffer is empty, the consumer must wait until the producer adds an item.
  - Consumer needs an item
  - # of item is represented by a semaphore *full*

The Bounded-Buffer Problem

- Producer-consumer problem with bounded buffer
  - Semaphores: full = 0, empty = n, mutex = 1;

  - Producer

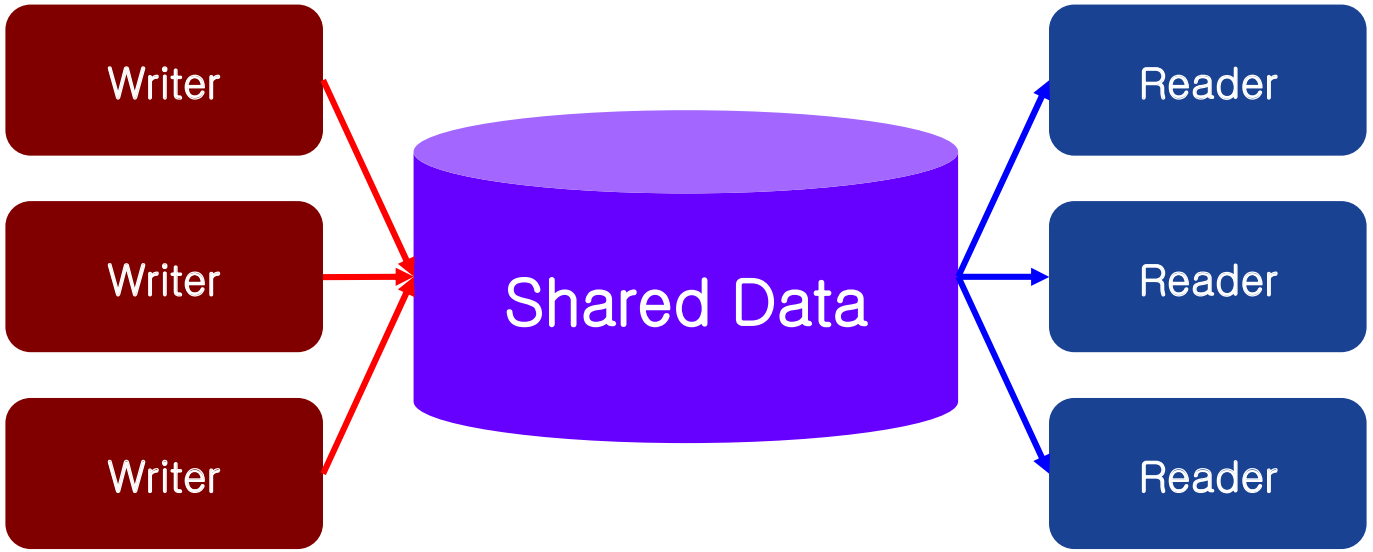
```
do {
        ...
    produce an item in nextp
        ...
    wait(empty);
    wait(mutex);
        ...
    add nextp to buffer
        ...
    signal(mutex);
    signal(full);
} while (1);
```

  - Consumer

```
do {
    wait(full);
    wait(mutex);
        ...
    remove an item from buffer to nextc
        ...
    signal(mutex);
    signal(empty);
        ...
    consume the item in nextc
        ...
} while (1);
```

The Readers-Writers Problem

- There are multiple readers and writers to access a shared data
  - Readers can access database simultaneously.
  - When a writer is accessing the shared data, no other thread can access it.



The Readers-Writers Problem

- Behavior of a writer
  - If a thread is in the critical section, all writers must wait.
  - The writer can enter the critical section only when no thread is in its critical section.
    - It should prevent all threads from entering the critical section.
- Behavior of a reader
  - If no writer is in its critical section, the reader can enter the critical section.
  - Otherwise, the reader should wait until the writer leaves the critical section.
  - When a reader is in its critical section, any reader can enter the critical section, but no writer can.
    - Condition for the first reader is different from the following readers.

The Readers-Writers Problem

- Shared data
  - semaphore mutex=1, wrt=1;
  - int readcount = 0;
    - # of readers in critical section

- Writer

```
wait(wrt);
    ...
writing is performed
    ...
signal(wrt);
```

- Reader

```
readcount++;
if (readcount == 1)
    wait(wrt);
    ...
reading is performed
    ...
readcount--;
if (readcount == 0)
    signal(wrt);
```