# The Readers Writers Problem

A database is to be shared among several concurrent processes. Some of these processes may want only to read the database, whereas others may want to update (that is, to read and write) the database. If two readers access the shared data simultaneously, no adverse effects will result. However, if a writer and some other thread (either a reader or a writer) access the shared object simultaneously, chaos may ensue.

This synchronization problem is referred to as the '*readers-writers problem'*. The readers-writers problem has several variations:

- **The First Readers-Writers Problem**
  It requires that no reader will be kept waiting unless a writer has already obtained permission to use the shared object.
  In other words, no reader should wait for other readers to finish simply because a writer is waiting.
- **The Second Readers-Writers Problem**
  It requires that, once a writer is ready, that writer performs its write as soon as possible.
  In other words, if a writer is waiting to access the object, no new readers may start reading.

A solution to either problem may result in starvation. In the first case, writers may starve and in the second case, readers may starve.

## Solution to The First Readers-Writers Problem

**Shared Data**

```
semaphore mutex, wrt;
int readcount;
```

**Initially**

```
mutex=1;
wrt=1;
readcount=0;
```

**The Structure of a Writer Process**

```
do
{
        wait (wrt);
        . . .
        //Writing is performed
        . . .
        signal (wrt);
} while (true);
```

**The Structure of a Reader Process**

```
do
{
        wait (mutex);
        readcount++;
        if(readcount==1)
```

```
            wait (wrt);
        signal (mutex);
        . . .
        //Reading is performed
        . . .
        wait (mutex);
        readcount--;
        if (readcount == 0)
            signal (wrt);
        signal (mutex) ;
} while (true);
```

- '*readcount'* is used to keep track of how many processes are currently reading the object.
- *'mutex'* is used to ensure mutual exclusion when the variable *readcount* is updated.
- *'wrt'* is used to ensure mutual exclusion among writers. It is also used by the first or last reader that enters or exits the critical section.

## Solution of The Second Reader Writer Problem

**Shared Data**

```
int readcount;
int writecount;
semaphore rmutex;
semaphore wmutex;
semaphore readtry;
semaphore resource;
```

**Initially**

```
readcount=0;
writecount=0;
rmutex=1;
wmutex=1;
readtry=1;
resource=1;
```

**The Structure of a Reader Process**

```
do
{
        wait(readtry);
        wait(rmutex);
        readcount++;
        if(readcount==1)
            wait(resource);
        signal(rmutex);
        signal(readtry);
        . . .
        //Reading is performed
        . . .
        wait(rmutex);
```

```
        readcount--;
        if(readcount==0)
                signal(resource);
        signal(rmutex);
}while(true);
```

**The Structure of a Writer Process**

```
do
{
        wait(wmutex);
        writecount++;
        if(writecount==1)
                wait(readtry);
        signal(wmutex);
        wait(resource);
                . . .
        //Writing is performed
        . . .
        signal(resource);
        wait(wmutex);
        writecount--;
        if(writecount==0)
                signal(readtry);
        signal(wmutex);
}while(true);
```

- '*readcount'* is used to keep track of how many processes are currently reading the object.
- '*writecount'* is used to keep track of how many processes are currently willing to write the object.
- *'rmutex'* is used to ensure mutual exclusion when the variable *readcount* is updated.
- *'wmutex'* is used to ensure mutual exclusion when the variable *writecount* is updated.
- *'readtry'* is used to ensure mutual exclusion between the reader processes as more than one reader process can simultaneously read the object.
- *'resource'* is used to ensure mutual exclusion among both readers and writers to database object.

## Questions asked in semester exam:

**Question:** Give the solution of Reader-Writer problem by using the concept of semaphore? [2015-2016] [10 Marks]

**Question:** Give reader/writer problem and its solution using semaphores. [2014-2015] [5 Marks]

**Question:** State the Readers/Writers Problem with readers having priority. Give solution of the problem using semaphores. [2012-2013] [10 Marks]

**Question:** Give the solution of Readers-Writers problem by using the concept of semaphore? [2008-2009] [10 Marks]