```
class ThreadContext {
 public:
  std::vector<void*> pending_reclaims;
  std::atomic<void*> *reservations;
  ThreadContext *next;
  const int num;

  ThreadContext(int _num, MemManager m): num(_num) {
    reservations = new std::atomic<void*>[_num];
    for (int i = 0; i < num; ++i)
      reservations[i] = nullptr;
    while (true) {
      next = m.head;
      if (m.head.compare_exchange_strong(next, this))
        break;
    }
  }
}
class MemManager {
 public:
  static thread_local ThreadContext *self = nullptr;
  std::atomic<ThreadContext*> head;
  ...
}
```

FIGURE 19.6 Data types to support hazard pointers with blocking reclamation.